AALTO UNIVERSITY
SCHOOL OF SCIENCE
Master's programme in Engineering Physics
Physics of advanced materials

Master's thesis

Digitising Historical Solar Intensity Maps for Metsähovi Radio Observatory

Sami Kivistö

August 9, 2018

Instructors: Juha Kallunki, Frederick Gent, Joni Tammi
Supervisor: Juho Kannala

Aalto University
School of Science
Master's Programme in Engineering Physics

Aalto University
School of Science

ABSTRACT
OF MASTER'S THESIS

Author: Sami Kivistö

Title:

Digitising Historical Solar Intensity Maps for Metsähovi Radio Observatory

Date: August 9, 2018 **Pages:** 120

Major: Physics of Advanced Materials **Code:** SCI3057

Supervisor: Juho Kannala

Instructors: Frederick Gent, Juha Kallunki, Joni Tammi

Metsähovi Radio Observatory has collected solar intensity maps since autumn 1978. The maps have been recorded with the main radio telescope at the observatory, which has a parabolic antenna with diameter 14 meters. The most common frequency is 37 GHz with beam diameter 2.4 arc minutes. Occasional observations have been conducted, for example, on 22 GHz and 77 GHz. The observational data were recorded on magnetic tapes or disks until 1987.

The original data is lost, but the maps have survived as mechanically rendered contour plots, some of which were also published as report series by former Helsinki University of Technology, Metsähovi Radio Research Station. In order to convert scanned images of the maps into a more usable format, I have written a software as part of my master's thesis during spring 2018. After appropriate image filtering, all the markings on the maps can be interpreted using the methods described in my thesis. Contour lines can be converted into a matrix form, which represents the observed intensity in a rectangular grid. This is achieved by solving a Poisson's equation accosiated with the problem. The method is tolerant for various map defects, such as broken contours.

In addition, I describe an algorithm for identifying bright and dim regions from these radio intensity map and show preliminary statistics based on the historical contour plots. The same algorithm is applicable to the modern maps as well, which are produced daily in Metsähovi.

Keywords: Sun, radio astronomy, historical records, contour maps, image recognition, Poisson's equation

Language: English

Aalto-yliopisto
Perustieteiden korkeakoulu
Teknillisen fysiikan maisteriohjelma

Aalto-yliopisto
Perustieteiden korkeakoulu

DIPLOMITYÖN
TIIVISTELMÄ

Tekijä: Sami Kivistö

Otsikko:

Auringon historiallisten kirkkauskarttojen digitointi Metsähovin radio-observatoriolle

Päiväys: 9. elokuuta 2018 **Pages:** 120

Pääaine: Kehittyneiden materiaalien fysiikka **Koodi:** SCI3057

Valvoja: Juho Kannala

Ohjaajat: Frederick Gent, Juha Kallunki, Joni Tammi

Metsähovin radio-observatorio on kerännyt Auringon kirkkauskarttoja syksystä 1978 alkaen. Kartat on kerätty observatorion pääradioteleskoopilla, jossa on 14-metrinen parabolinen antenni. Yleisin käytetty taajuus on 33 GHz, joka tuottaa halkaisijaltaan 2,4 kaariminuutin levyisen keilan. Yksittäisiä havaintoja on tehty mm. taajuuksilla 22 GHz ja 77 GHz. Havaintodata on tallennettu magneettisille nauhoille tai levyille vuoteen 1987 asti.

Alkuperäinen data on kadonnut, mutta kartat ovat selvinneet mekaanisesti piirrettyinä korkeuskäyräkarttoina, joista osa on julkaistu silloisen TKK:n Metsähovin Radiotutkimusaseman sarjakokoelmissa. Jotta karttojen skannatut kuvat saataisiin paremmin hyödynnettävään muotoon, olen kirjoittanut ohjelmiston osana diplomityötäni keväällä 2018. Sopivan kuvansuodatuksen jälkeen kaikki kuvan merkinnät voidaan tunnistaa työssäni kuvatuilla menetelmillä. Korkeuskäyrät voidaan muuntaa matriisimuotoon, jossa havaittu kirkkaus on esitetty suorakulmaisessa verkossa. Tämä onnistuu ratkaisemalla ongelmaan liittyvä Poissonin yhtälö. Menetelmä sietää erilaisia karttavirheitä, kuten katkenneita korkeuskäyriä.

Lisäksi kuvailen algoritmin, jolla tunnistetaan radiointensiteettikartoista kirkkaat ja himmeät alueet, sekä esittelen alustavia tilastoja historiallisista korkeuskäyräkartoista. Sama algoritmi soveltuu myös uusiin karttoihin, joita tuotetaan Metsähovissa päivittäin.

Avainsanat: Aurinko, radiotähtitiede, historialliset tallenteet, korkeuskäyräkartat, kuvantunnistus, Poissonin yhtälö

Kieli: englanti

Preface

I want to thank Maarit Käpylä and everyone else in the ReSoLVE Astroinformatics group for valuable commentary and inspirational feedback during writing this thesis. Especially my instructor Frederick Gent has given me numerous comments considering written language and science. I also want to thank Eija Tanskanen and the folks in the ReSoLVE MAGNETIC team for sharing the enthusiasm and optimistic attitude with me. When I was searching through the list of input and output files and looking for examples to beam at the screen, Eija announced that we will produce a science paper out of this right away, and personally started writing it while I was speaking. I want to thank the people in Metsähovi, including my instructors Juha Kallunki and Joni Tammi, and also Merja Tornikoski for all the positive and constructive feedback and encouragement. My supervisor Juho Kannala from the Department of Computer Science also deserves my graditude for giving wise advice while I was writing this thesis. Special thanks go to Ilja Pippa and Katariina Korhonen for the independently viewing and categorising the output maps produced in this thesis.

I also want to thank my beloved Mari for her natural enthusiasm towards my work as well as for bringing balance in my life and standardising my working hours. Our bunny, Otto, also reserves my graditude for crunching numbers and everything.

Sami Kivistö,

Kerava, August 9, 2018

Contents

1 Introduction

Metsähovi Radio Observatory is a separate institute of Aalto University School of Electrical Engineering. It is the only astronomical radio observatory in Finland, located in Kirkkonummi, with ca. $30km$ straight distance from Helsinki. Metsähovi has performed solar observations irregularly since 1976, with equipment suitable for frequencies 5...120 GHz [1]. The main radio telescope is covered by a radome, which protects it from wind, rain, and solar heat. That enables solar observations, which is a rare ability for radio observatories around the world. Without the radome, the reflective surface of the dish would concentrate solar light and incinerate the receiver, and the dish would be subject to drastic temperature changes and gradients. Metsähovi strives to produce at least one map every day [2], also during winter [3]. Thanks to the high latitude, the observatory is dedicated to continuous solar observations for at least 12 hours per day during summer months.

Regular solar observations began in autumn 1978 with sweeping the Sun using the main dish. While sweeping, the antenna is rotated along the equator. After each line is completed, the antenna is returned to its original right ascension value, and the declination is changed in order to initiate the next sweep line. This allows us to construct intensity maps. The signal resulting from the sweeping has originally been stored on magnetic tapes or disks in a digital form [4]. Later, the data were processed in a computer program, which performed interpolation and controlled a mechanical XY plotter, drawing contour graphics from of the data [5]. An example is given in Fig. 1a.

Unfortunately, the magnetic tapes have later been lost, and the early data are only available in these plots. The paper maps were scanned earlier into digital images in Portable Document Format (PDF). This includes ca. 1000 maps from the years 1978 to 1987. Observations have since continued with constantly developing technology, now extending 40 years. The present maps are digitally rendered, as in Fig. 1b. With this work, we are able to extend our solar observational history from autumn 1978 into present, so that digital radio intensity maps can be compared between different solar cycles.

There are ca. 1000 maps in this contour plot format, of which the oldest is from September 6, 1978, and the newest from September 4, 1987. Maps from the years 1978 to 1986 are published in Metsähovi report series [5–8]. The maps from the year 1987 are printed with matrix printer technology [9], but fortunately contour plots were also available. The data from the year 1988 are only available in matrix printer format [9], which will require heuristics in order to be correctly interpreted.

When considering the contour plots, there are mainly two types of maps:

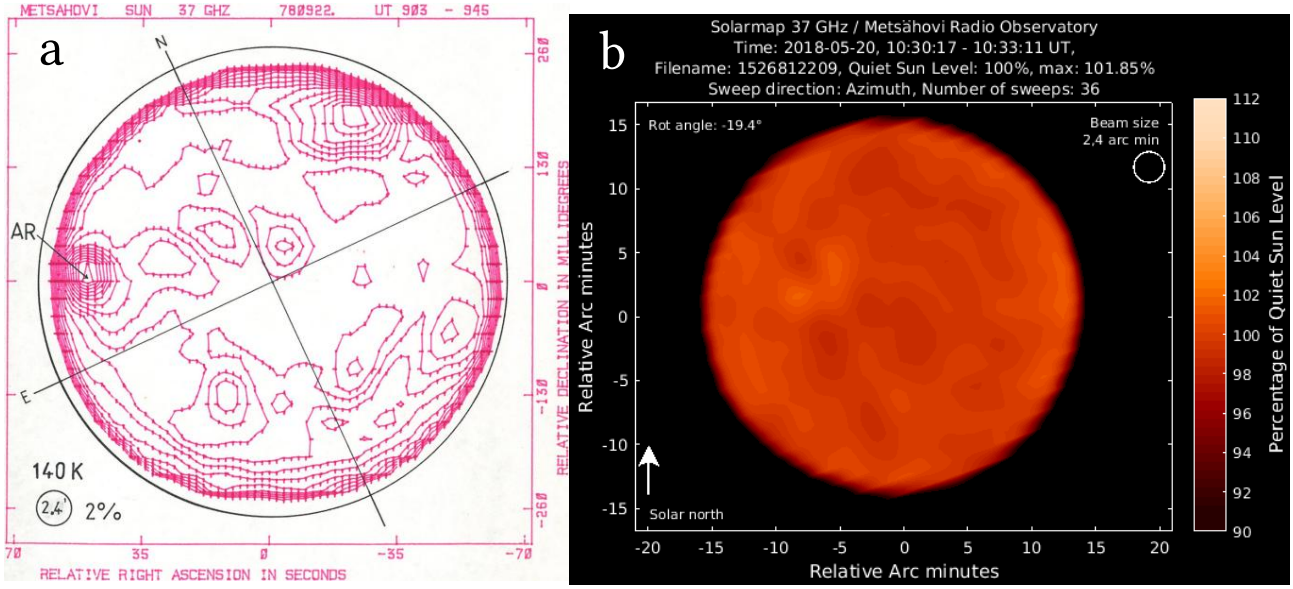

Figure 1: a) Historical contour plot from the year 1978. b) Modern map from the year 2018.

- Full solar maps
- Partial maps from especially interesting regions

Sometimes several full solar maps and partial maps have been rendered from the same observational data. This is evident since a map always contains two timestamps, which refer to when the observation started and when it was finished.

Solar research is based on long time observational data, while the methodology and techniques used will vary from decade to decade. Historical records need to be calibrated and converted into a common format, so that they can be reliably compared. When doing this interpretation, there is always a source of error and a better way to perform the digitisation. This might involve fine-tuning some of the digitisation parameters, or even implementing new algorithms in order to treat some artifacts. Every time such an improvement is discovered, the data have to be reprocessed, thus multiplying the cost of one time digitisation. It quickly comes necessary to fully automate the process.

1.1 The software

The process of writing the digitisation code has been likewise iterative, and the first achievement was to successfully process one particular map in December 2018. Additional adjustments were required in order to process a few more maps, so that the parameter space has been constantly expanding. This has required a framework for optimising the parameter space efficiently. Since March 2018, when most of the maps were successfully processed, also some preliminary statistical analysis of the results has been made. All the computational tasks were completed with a standalone C++ code compiled for 64 bit Linux environment. The particular executable and the associated code package is here called the *sunmap*, which was developed and is published along with this thesis. The package is available in [10].

The code was compiled with G++ version 5.4.0 [11]. When compiling, only the basic system libraries are required: *libc* [12], *libm* [13], *libgcc* [14], *libstdc++* [15], and *linux-vdso* [16]. The code should compile and run without complications on a typical Linux workstation. The code utilizes *Bash* [17] scripts for feeding the sometimes long list of command line parameters into the *sunmap* executable. These scripts are also used for optimising parameter configurations and utilising multiple workstations and processes in parallel as well as for demonstration purposes. For converting between different image formats, Linux *netpbm* [18] tools have been used extensively. Most of the graphs are drawn using *gnuplot* [19], while some pictures are drawn solely with the methods included in the *sunmap* package. For performing the analysis of active regions, there is an ad-hoc executable, called *follow*, which performs the necessary calculations.

One contour map is typically processed in 1 min on a typical workstation. When optimising the parameter configuration for a particular map, several hundred runs are made. Since the workload can be distributed over several workstations, the task of re-processing the original maps will require a few hours, and it was done several times during development. This is achieved by storing the input maps in a network filesystem, so that all the workstations with separate processes may access them and process their share in an embarrassingly parallel manner. Most time-consuming part appears to be the dimensionality reduction associated with the Poisson solver. Another intensive task is the merging of lines and circles for the layout detection. Any further optimisation should thus concentrate on these two stages.

The code was developed as part of this master thesis and mostly from scratch, although many accessories and utility functions originate from previous course assigments that I have done as part of my bachelor and master studies. The work continues by further improving the map quality as well as by unified calibration with newer data sets. New approaches, such as neural networks for contour and character recognition, are likely to be experimented with the subsequent versions, as the code would benefit from better adaptivity in various parts of the data flow. These techniques will find additional use for other historical documents with contours. Plans already exists for digitising solar magnetograms, which were collected in Mount Wilson and Palomar Observatories during 1959 to 1966 [20]. An example is shown in Fig. 2. These algorithms are suitable for digitising geographic contour maps as well.

Also the statistics tools developed in this project find more general use. They are able to detect bright and dim regions in the modern Metsähovi maps as well. Before the maps can be processed, the sparse antenna tracks need to be interpolated with a suitable algorithm, since the region detection is based on rectangular matrix input. The interpolation has to be tolerant for antenna wiggling during

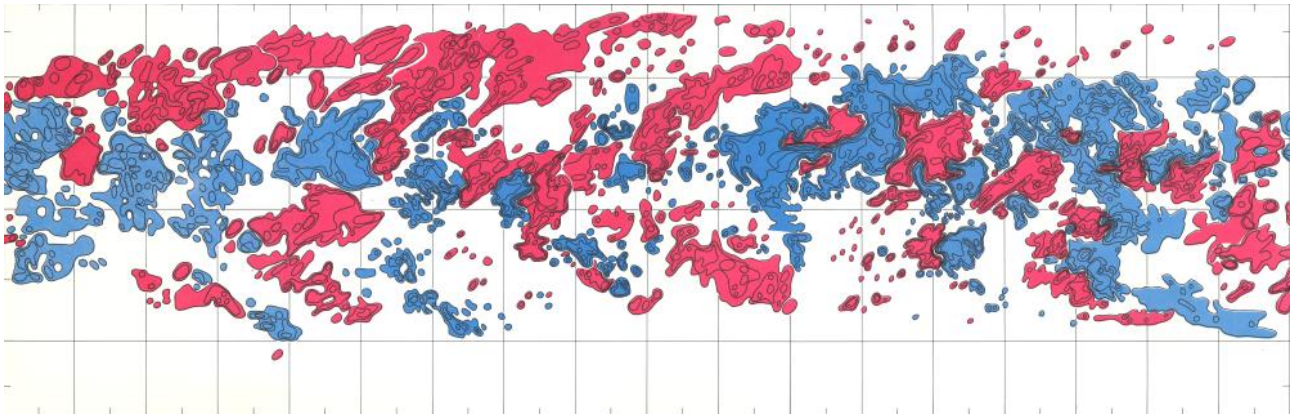

Figure 2: *A solar magnetogram from Mount Wilson or Palomar observatory. Obtained with collaboration from ReSoLVE Center of Excellence [21].*

sweep, although the wiggling has already been fixed for the most recent maps.

1.2 Objectives of this work

The main task was to recover the lost information, that originally was used for producing solar intensity contour plots for the years 1978 to 1987. As the solar maps were already scanned, the next step was to distinguish meaningful features from these map images. The minimum requirement was to detect relevant solar brightness features, but being able to obtain the complete intensity distribution over the whole disk would be preferable.

In this thesis, an algorithm is developed for interpreting these contour plots. They are converting into a modern data format, which is directly utilisable and both human and machine readable. The program has a wide input parameter space, which allows the original contour plots to be reprocessed in case additional defects are observed in later scientific work.

One of the most central features is the ability to convert contour plots into intensity maps. This is accomplished by first solving a Poisson's equation with the analogy that contours consist of electric dipoles, and the intensity is an electric potential produced by those dipoles. We are able to intepret even broken contour maps, since the missing parts of contours can be represented as perturbations to a second Poisson's equation. This second Poisson's equation is based on complete contours, which we initially do no know. Beyond the missing parts, the perturbation is a harmonic function with absolute gradient disminishing at a distance. We eliminate the perturbation by means of substitution, and iteratively obtain new solutions to the first equation, with less degrees of freedom after each iteration. Finally we will reach the above mentioned second Poisson's equation and have connected stray ends of contours in a meaningful way. This allows us to produce realistic output even from degraded input.

This program is able to analyse the radio intensity distribution for bright and dim regions, which significantly deviate from the Quiet Sun Level (QSL). The code

produces hierarchical output, so that fine detail and structure of those regions is also obtained. For example, we can use the obtained hierarchical database to classify radio brightenings.

The program will calculate the center of mass $\boldsymbol{r}_{\mathrm{CM}}$ for each region, which consists of cells on $N \times N$ rectangular grid, each cell having an index $i = 0, ..., N^2 - 1$. Its location on the sky in equatorial coordinates is specified as $\boldsymbol{r}_i$, and the intensity is v_i. Center of mass is averaged for absolute difference of cell brightness to the Quiet Sun Level v_{QSL} as:

$$\boldsymbol{r}_{\mathrm{CM}} = \frac{\sum_i \boldsymbol{r}_i \left| v_i - v_{\mathrm{QSL}} \right|}{\sum_i \left| v_i - v_{\mathrm{QSL}} \right|}. \tag{1}$$

The regions are analysed in heliographic Carrington coordinates, where we can obtain statistics of their time evolution. This particular branch in the code is also suited for modern maps, such as in [3] and in Fig. 1b. In the near future, we will process all of the more than 40 years of Metsähovi solar intensity data, and extensively analyse the behaviour of radio brightenings and dimmings.

1.3 The Sun

The Sun consists of mainly hydrogen and helium and originates from a cloud of interstellar gas $\approx 5 \times 10^9$ years ago. As this cloud collapsed due to gravity, it was subject to adiabatic heating, until at some point the temperature was sufficient for fusion reactions. The resulting excess heat prevents further collapse. The energy is transported first by radiattion and later mainly by convection. Between the radiative core and the convective zone is a layer called the tachocline (Fig. 3). The convective zone rotates differentially to the core, as convection continuously redistributes angular momentum [22]. The convective zone ends at the photosphere, which emits the visible light seen on Earth. Beyond the photosphere is chromosphere and corona.

Hot objects emit electromagnetic radiation at wide spectral ranges. This results from various physical processes, such as collisions and a dense network of broadened atomic and molecular transition lines. These transfer energy between the dynamic particles and the electromagnetic field. As an example of such, an accelerating charged particle emits electromagnetic radiation due to free-free transions, which is known as bremsthrahlung. When the coupling between particles and the electromagnetic field is strong, so that the electromagnetic field is in thermal equilibrium with the material, the emission will follow Planck's law (see Appendix A). [22] The microwave observations in Metsahovi solar intensity maps are based on very low frequencies, in terms that

$$h\nu \ll k_{\mathrm{B}} T \tag{2}$$

where h is the Planck's constant, $\nu = \frac{\omega}{2\pi}$ is the observed frequency, k_{B} is the Boltz-

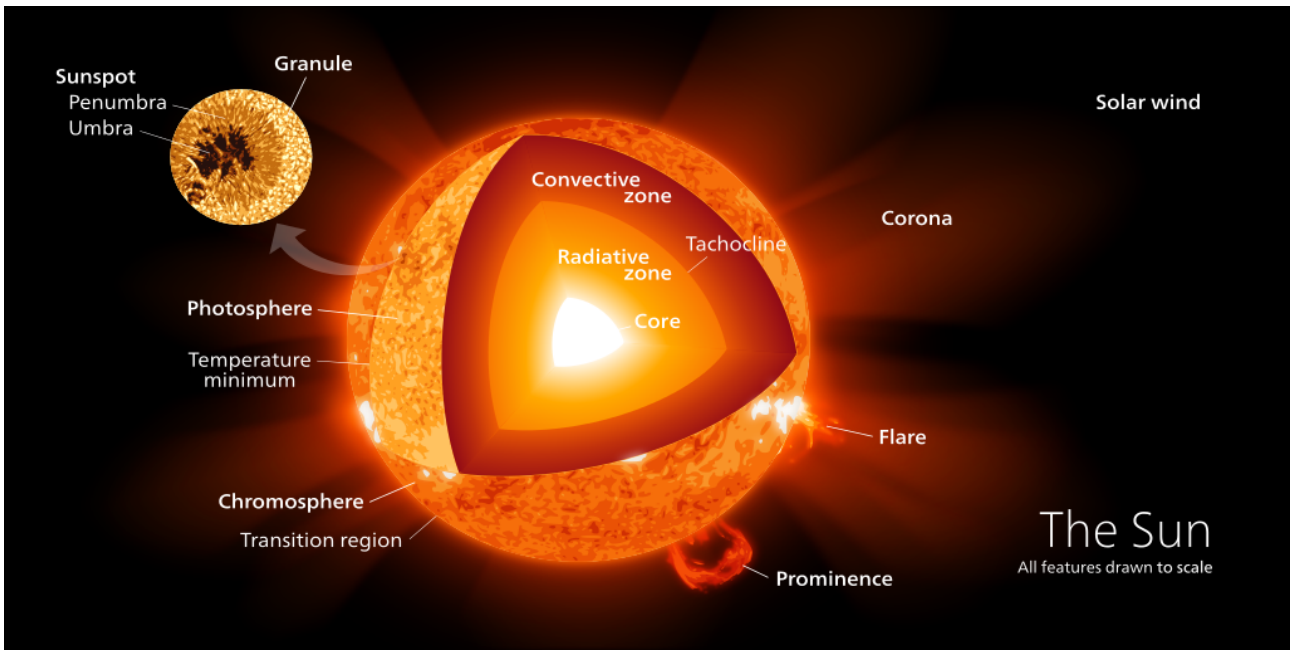

Figure 3: *Structure of the Sun. Picture from Kelvinsong / Wikimedia Commons [23].*

mann's constant, and T is the observed solar temperature. Based on Rayleigh-Jeans approximation, which is obtained from the Planck's law for low frequencies, we can express the signal intensity in temperature units.

Microwave emissions are often related to magnetic activity. Free charged particles adopt circular paths in a magnetic field, due to Lorentz force. This acceleration produces electromagnetic radiation, which is known as gyroemission [24]. Collecting observational data considering solar magnetic activity is important, since the scientific community needs to be able verify, compare, and falsify various solar theories. Solar activity and space weather also affects many aspects in society, such as power grids, communication and space activity. [21]

In general, electromagnetic waves can not propagate if their frequency is higher than the plasma frequency ω_p in the medium. That depends mainly on the density of free electrons n_e [22] as:

$$\omega_\mathrm{p} = e\sqrt{\frac{n_\mathrm{e}}{\epsilon_0 m_\mathrm{e}}}, \tag{3}$$

where e denotes the elementary charge and ϵ_0 is the electric permittivity of vacuum. The plasma contains also positive ions, which are heavier than electrons by three orders of magnitude, so that their contribution to the plasma frequency can be neglected. Electrons absorb energy from a propagating electromagnetic wave and convert it to thermal energy via collisions. In a classical Drude model [25], the mean time between collisions is τ. Then the electromagnetic wave with angular frequency ω will have complex phase velocity $\frac{c}{\tilde{n}}$ determined by the complex

relative permittivity of the material, ϵ_r, as [26]:

$$\epsilon_\mathrm{r} = 1 - \frac{\omega_\mathrm{p}^2}{\omega^2 + \mathrm{i}\omega/\tau}. \tag{4}$$

The material has refractive index $\bar{n}$, which contains a real part $\bar{n}_\mathrm{R}$ and an imaginary part $\bar{n}_\mathrm{I}$:

$$\bar{n} = \sqrt{\epsilon_\mathrm{r}} = \bar{n}_\mathrm{R} + \bar{n}_\mathrm{I}\mathrm{i}. \tag{5}$$

The imaginary component results in decaying amplitude for the wave which propagates the medium. We can relate the imaginary component into the optical thickness l of the medium as:

$$l = \frac{c}{2\omega\bar{n}_\mathrm{I}}, \tag{6}$$

where c is the speed of light. When the radiation has propagated a distance z, its original intensity I_0 has dropped to $I(z) = I_0 e^{-z/l}$.

The convective zone works as a thermal resistor, so it is obvious that the temperature is higher with depth. The solar atmosphere is more ionized when the temperature is higher, and also thicker due to gravity, so that only above the photosphere the Sun is generally opague to visible light. Visual light is thus generally able to propagate from the photosphere into outer space, while only some characteristic wavelengths are absorbed and emitted in the chromosphere. [22]

The features observed at microwave frequencies also originate from the chromosphere [27]. As the plasma frequency decreases with height, the chromosphere becomes opague to lower frequencies of electromagnetic radiation. By selecting the radio frequency for observation, we can probe different layers in the chromosphere. In the corona the temperature in turn increases with height, mainly due to various reconnection processes [24]. The corona heats the upper layers of the chromosphere, as is studied e.g. in [28]. Shorter radio wavelenghts show higher signal temperatures, as shown in Fig. 4.

Sunspots are photospheric regions of strong magnetic field and decreased temperature [22]. They are sometimes accompanied with radio brightnenings and high energy radiation emissions. For an example, the Sun is shown in three different parts of the electromagnetic spectrum in Fig. 5. Two particular regions in the northern hemisphere coincide in all three wavelengths, although, due to contrast and resolution issues, they are barely distinguishable in visible light (Fig. 5b) as sunspots. Sunspot numbers follow the activity cycle of ca. 11 years (Fig. 6) during which they wander from the middle latitudes towards the equator (Fig. 7). A similar pattern is observed from the statistics of especially bright radio intensity regions, the radio brightenings (Fig. 8) [27].

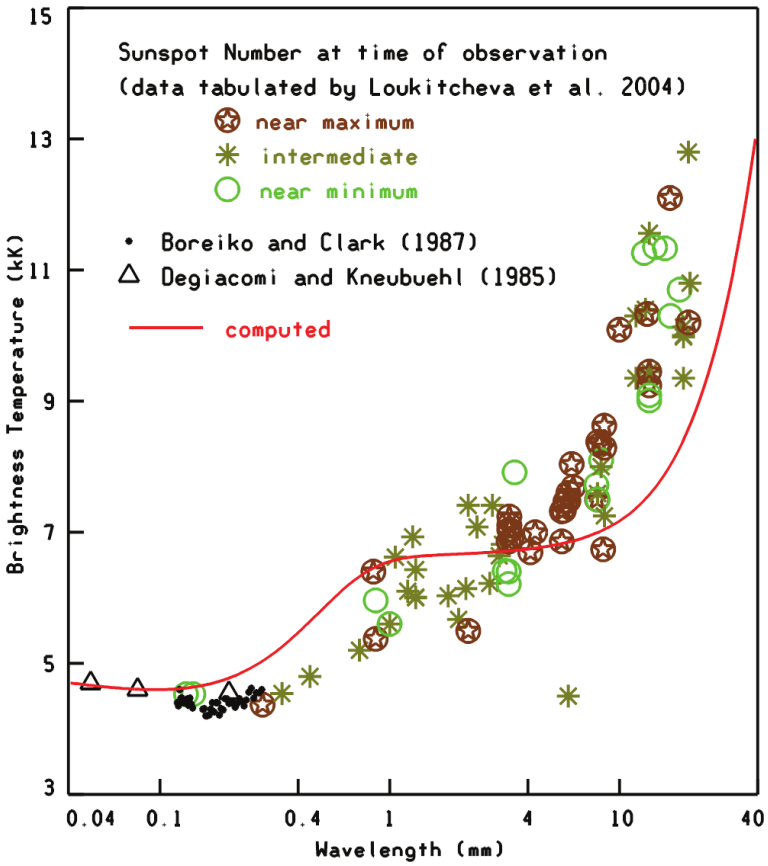

Figure 4: *Measured and computed solar brightness temperatures at microwave and far infrared wavelengths. Picture from Eugene H. Avrett and Rudolf Loeser [28].*

Various regions are constantly oscillating in frequencies of several differents orders of magnitude, ranging from seconds to days [24]. These oscillations are quasi-periodic, meaning that they do not have a fixed frequency but rather consitute a wider band or a peak of finite width. In longer scale, the solar magnetic activity follows a cycle of ca. 22 years, so that the solar magnetic dipole switches to opposite polarity in approximately every 11 years. The science community is trying to explain this phenomenon as a self-starting dynamo [21].

The dynamo will not work without a suitable three-dimensional plasma flow configuration. In a self-starting dynamo, a small perturbation in the electromagnetic field will start growing exponentially, until the growth of magnetic field is restricted by Lenz's rule. The Lorentz force $\boldsymbol{j} \times \boldsymbol{B}$ then starts affecting the flow configuration. [22]

The exact configuration required for the self-starting dynamo is not perfectly understood, but the flux-transport Babcock-Leighton dynamo theory places it at the tachocline, where the differential rotation transforms polar magnetic field into a toroidal field. Turbulent convection transports the toroidal field lines into the photosphere, where it appears as sunspot groups or pairs, which will get tilted as they reach the equator. As the sunspot groups decay, their flux is transported to the polar regions in meridional flow. This theory dates to Babcock (1961) and Leighton (1964). [33,34] Yet there is a different theory that explains the formation

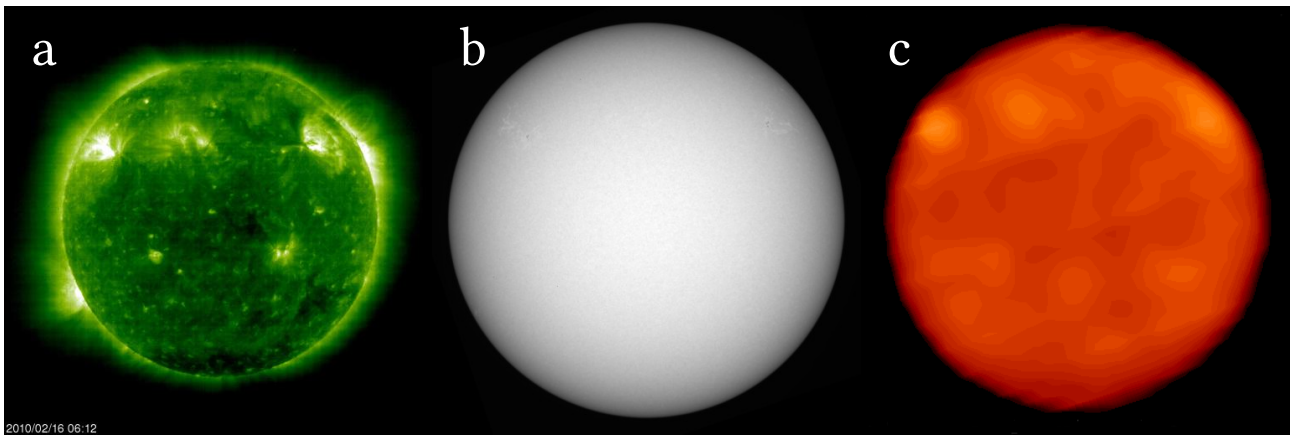

Figure 5: Sun at different wavelengths on February 16, 2010. In all pictures, the solar north axis of rotation points upwards. There are two features on the northern hemisphere. a) At 19.5 nm by SOHO Extreme ultraviolet Imaging Telescope (EIT) [29]. The features are clearly visible with detail. b) At blue continuum centered at 409.4 nm by Precision Solar pHotometric Telescope [30]. The features appear as sunspot groups, but are barely visible with this resolution. c) Intensity at 8 mm by Metsähovi Radio Observatory [3]. The features are observed as bright regions.

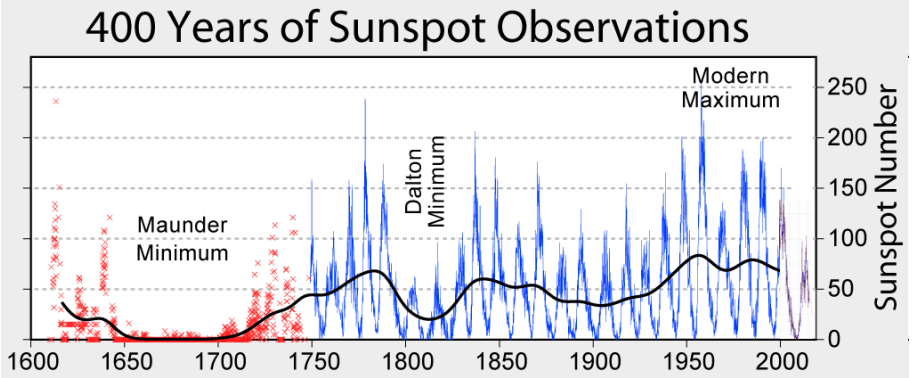

Figure 6: 400 *years of sunspot number observations aggregating various historical observation sources. Picture from Robert A. Rohde / Wikimedia Commons [31].*

of sunspots and the toroidal field to be far more localised outcome of turbulent transport processes just below the surface. This is known as the near-surface shear dynamo or the distributed dynamo. [33, 35]

An important phenomenon present is the α effect by Parker (1955) as well as Steenbeck, Krause, and Rädler (1966). As a rising parcel of convective plasma expands in the northern hemisphere, it acquires left-handed helical component in the presence of rotation, due to Coriolis force. The same is true for a sinking parcel, and the helicity is opposite in the southern hemisphere. This will twist the toroidal magnetic field back into polar field. [22, 34]

1.4 Metsähovi radio telescope

A telescope is an optical instrument which forms a finite image from a distant object. For electromagnitic radiation in radio frequencies, this is practically achieved using reflective surfaces, so that a plane wave from a particular angle will be focused to a particular point.

In a Cassegrain type telescope, we have a hyperbolic mirror as a secondary reflector, with focal points P and S, and a common optical axis. P is also the

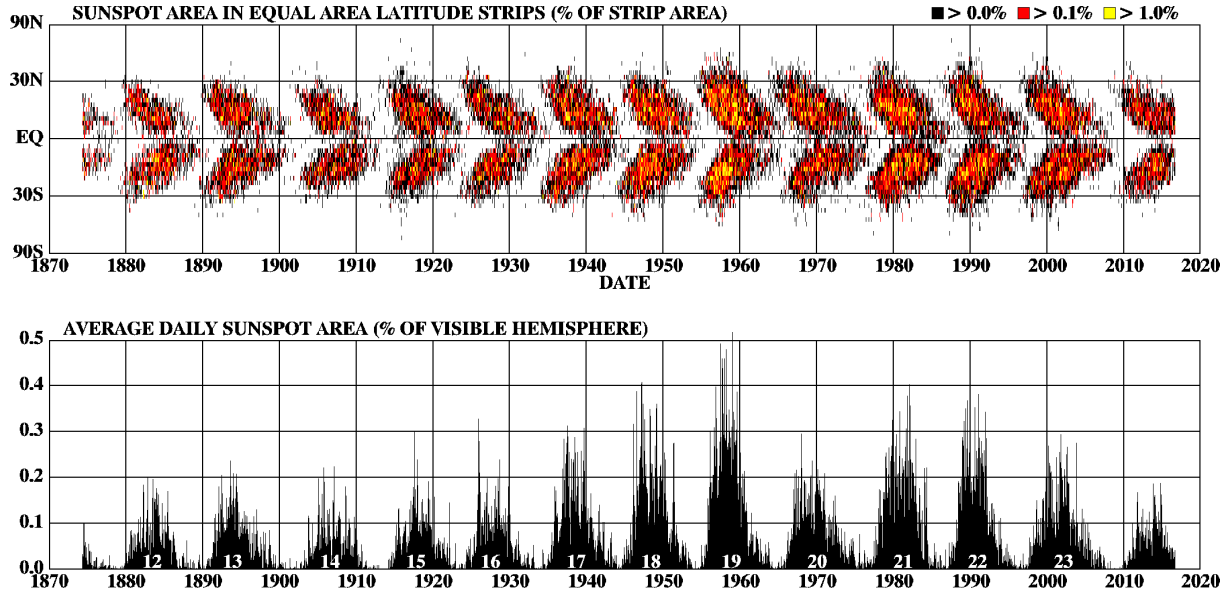

Figure 7: *Detailed observations of sunspots obtained by the Royal Greenwich Observatory. In the beginning of each solar activity cycle, sunspots are first seen at middle latitudes. Later, they appear closer to the equator. Each cycle lasts approximataly* 11 *years. Picture from David Hathaway / NASA / ARC [32].*

focal point of the parabolic mirror, so that incoming radiation is finally focused at S. This is close to the centre of primary parabolic reflector, where is also the receiver (Fig. 9a). Metsähovi has a Cassegrain type telescope with parabolic primary reflector diameter 13.8 m (Fig. 9b).

1.5 Historical solar intensity maps

All the maps from the years 1978 to 1987 have been recorded in sweep mode, where the antenna direction changes steadily in right ascension while the declination is kept constant. After each sweep, the antenna was returned to the original right ascension and the declination was changed by small amount. Then a new sweep was carried out with slightly different declination. Modern map data have the antenna position constantly recorded, and this allows measurements also while the antenna is returning and travelling backwards in right ascension. Measurements can also be done in tracking mode with the antenna following a particular region on the Sun, such as in [37].

The radio frequencies are listed in table 1.5, which includes also the number of successfully processed individual related to this thesis. Most data is recorded with 36.8 GHz.

The obtained data was mechanically rendered using either of two Hewlett-Packard plotters HP 9862A and HP 9872B (Fig. 10), as based on email conversation with Silja Pohjolainen. The archived sheets were later scanned on Portable Document Format files, which contain colour information and good resolution, so that line width corresponds to ca. 8 pixels.

The map consists of a square box. There is a title line on top of the box, and the

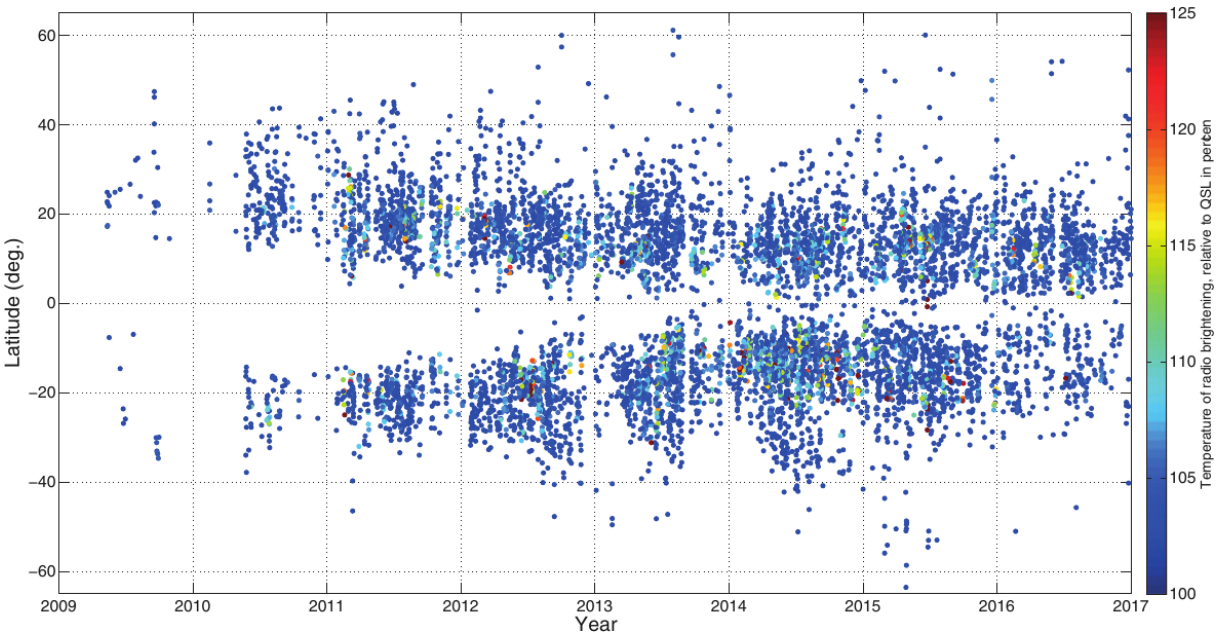

Figure 8: *Modern observations of radio brightenings, recorded in Metsähovi. Picture from Kallunki et al. [27].*

Frequency (GHz)	Wavelength (mm)	Telescope beam $size(\text{arc min})$	Estimated quiet Sun level (K)	Processed maps in 1978..1987
11.6	25.9	7.6	12000	5
22.2	13.5	4.0	9000	48
36.8	8.3	2.4	7800	414
75				1
77.1	3.9	1.2	7250	29
81				9
87	3.5	1.0	7200	1

Table 3: Radio frequencies used in Metsähovi Radio Observatory. Data from [9].

title line mentions Metsähovi and the observable 'Sun', as well as the date and begin and end time of observation. From left to right is decreasing right ascension, and from bottom to top is increasing declination. If the map features the whole Sun, there is typically a circle drawn on top of the map (Fig. 11a). The circle then has directional markings for solar axis and equator. As the Sun rotates, features travel from east to west on the solar disk. Partial maps have the rotational axis and the equator marked with lines on top of the map (Fig. 11b).

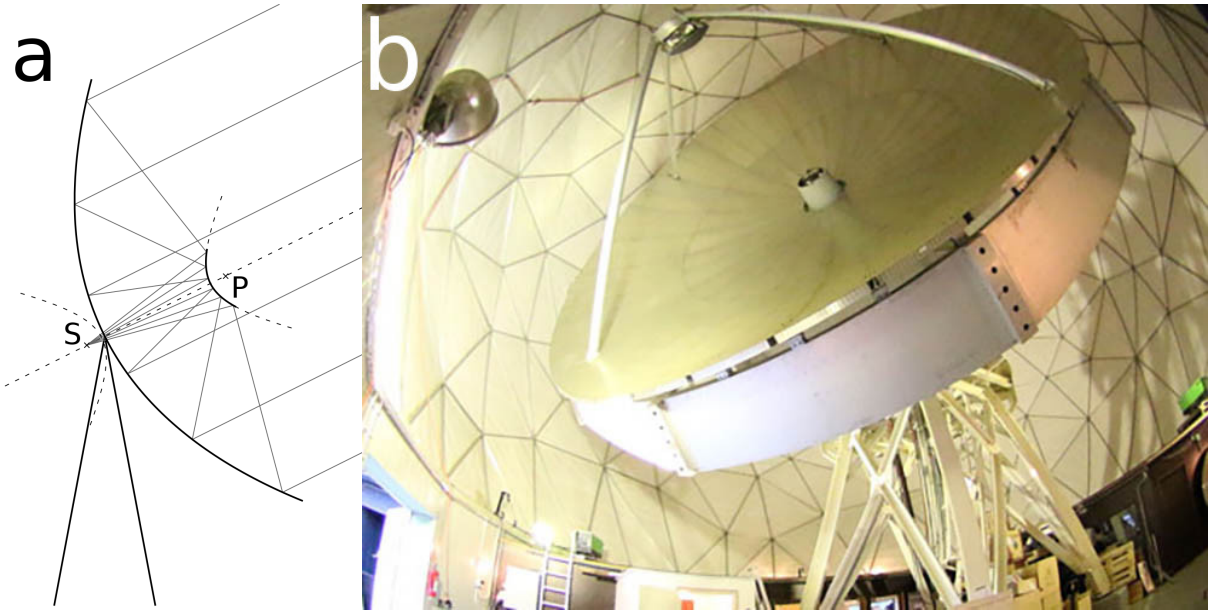

Figure 9: a) Radio waves are concentrated using a large paraboloid primary reflector and a small hyperboloid secondary reflector. The focal points of the secondary reflector are P and S, of which P is also the focal point of the parabolic reflector. The geometric hyberbolic sections, which define the secondary reflector, are drawn as extending dashed curves. b) Metsähovi main dish, the RT14. Photo from Metsähovi homepage [36].

Figure 10: Mechanical plotters HP 9862A and HP 9872B. Pictures from HP memory project [38].

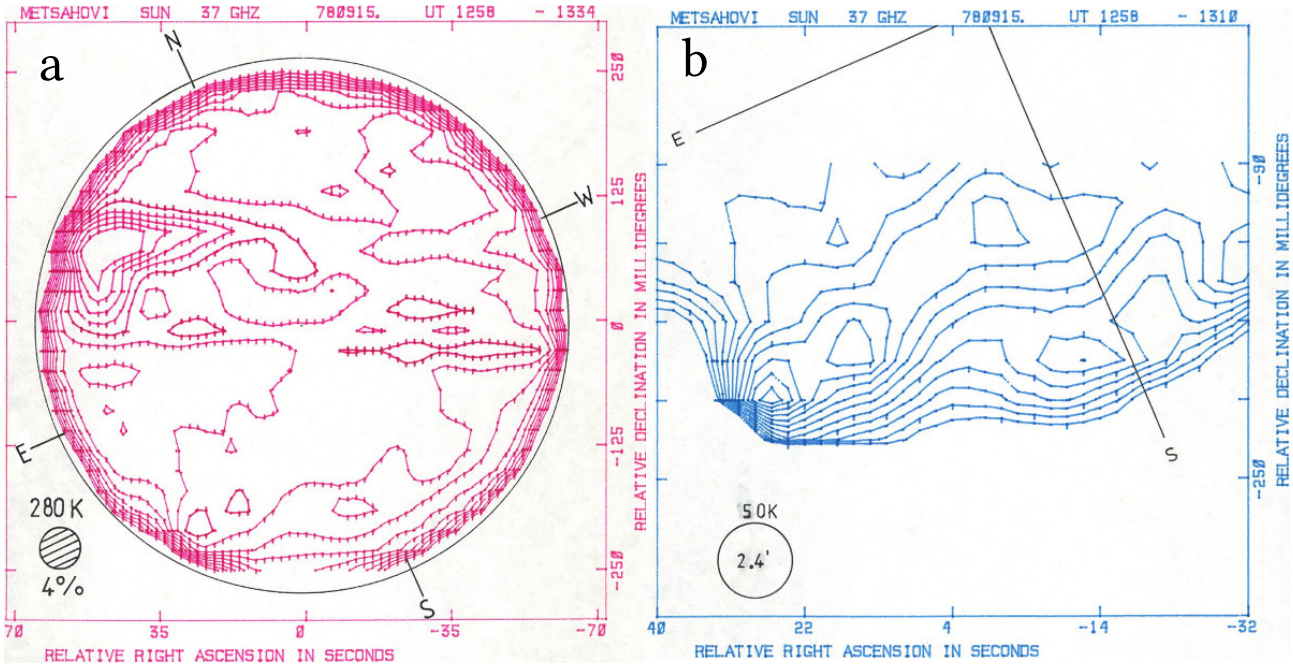

Figure 11: Complete solar disk (a) and a partial map (b) rendered from the same observational data.

2 Methods

In order to convert the scanned solar intensity maps into a useful digital format, the contour lines need to be detected. All markings related to the layout have to be detected, since otherwise they would be falsely identified as contour lines. This is not always obvious, since the maps usually have a circle representing the visible solar disk, superimposed with the contours. This large circle is hand-drawn with a pair of compasses after plotting the contours mechanically. On greyscale images, the only distinction that tells contours from a compass circle is the precise circular shape of the latter.

Often the solar disk is not fully visible, which results in broken contours (Fig. 12 and 13). Contour lines are concentrated at the terminator of the visible solar disk, since this is where the intensity drops drastically when advancing away from the centre. The terminator is the perimeter of the visual disk, where the tangent of the apparent spherical surface is incident with the light of sight from the observer.

The part of the visual disk near the terminator constitutes the limb, in which the apparent surface is observed at a low angle, thus leading to projection effects. This is useful when observing prominences [39], since we will get radial information from them as they pass the solar limb [40]. In the solar maps, these events are observed as brightenings at the dim, which can be problematic to interpret automatically. The contours often overlap outside the rim and flow backwards at the rim (Fig. 14). As we fallback all the rim contours to flow counterclockwise, we will run into trouble if the rim distance threshold is set too broad. The threshold can be adjusted with the parameter *perimeter_width* (Appendix B.24).

Overlapping contours are also common around central bright regions and may result from unsatisfactory rendering (Fig. 15). The algorithms must be able to treat broken contours either by suitable cropping of the measurement area or by connecting them. Ideally, contours should be complete and either make loops or end at the map boundary.

The outline of the program is presented in Fig. 16. The image contains various dark features, such as contour lines, and they can be distinguished through the brightness information in the image. We will choose a suitable threshold, which may depend on the location. Brightness value below the threshold is considered as a feature. We will tag suitable dark points and connect them into paths. After the layout is detected, we are left with a set of contours and and appropriate coordinate system.

We aim to construct scalar intensity values within the coordinate system, with the help of the contour lines and the fact that intensity is zero outside the solar

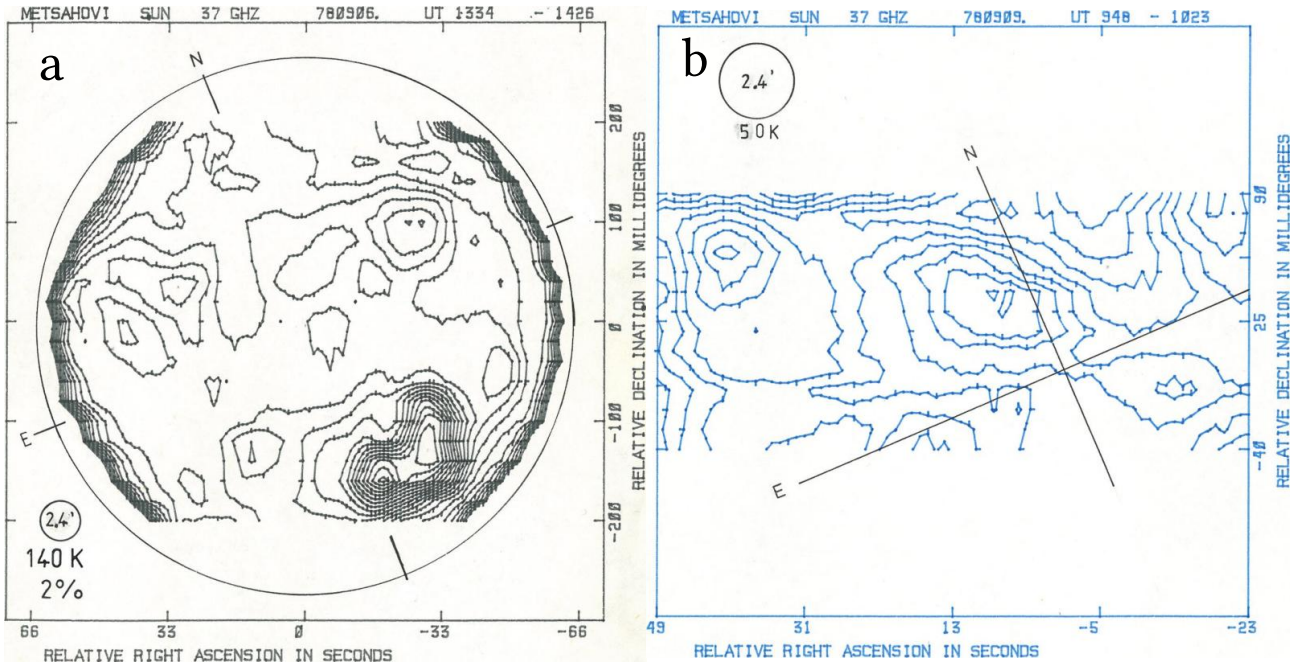

Figure 12: Cropping leaves broken contours. a) Map has limited number of equatorial sweeps and has to be cropped, while most the solar disk is partially visible. b) Contours extend beyond the boundary of the box, and no solar disk terminator is visible.

disk terminator. This is achieved with a Poisson solver. Since the contours are often broken, the error arising from missing contour segments is treated as a perturbation to an intact Poisson's equation. The perturbation is eliminated with appropriate substitutions between adjacent grid points. After a reasonable amount of substitutions to eliminate some perturbation, we are left with another Poisson's equation with reduced degrees of freedom. When this is again solved, we can continue with substitutions. Finally we will end up with a solution where broken contours do not cause wide peturbations.

Once we have an intensity map, we will determine the Quiet Sun Level (QSL) from statistics. Thresholds for dim and bright regions as well as the effective Quiet Sun Level can also be given from command line using parameters *QSL_contour_min*, *QSL_contour_max*, and *QSL_contour_mid*, respectively. Regions outside these boundaries are considered either dim or bright. The regions can be nested, as they are mentioned in the Java Script Object Notation (JSON) [41] output of the *sunmap* program. It is also possible to input modern maps into the data flow.

Historical solar maps need contour detection, after which the intensity values are obtained by solving Poisson's equation. Modern antenna samples are recorded in horisontal sweeps on regular time interval of e.g. 40 ms. Since they are asynchronous, they need to be interpolated and resampled on a rectangular grid. Since the output format is flexible, additional interpolation is possible when generating the final output. Identified features are reported in JSON format.

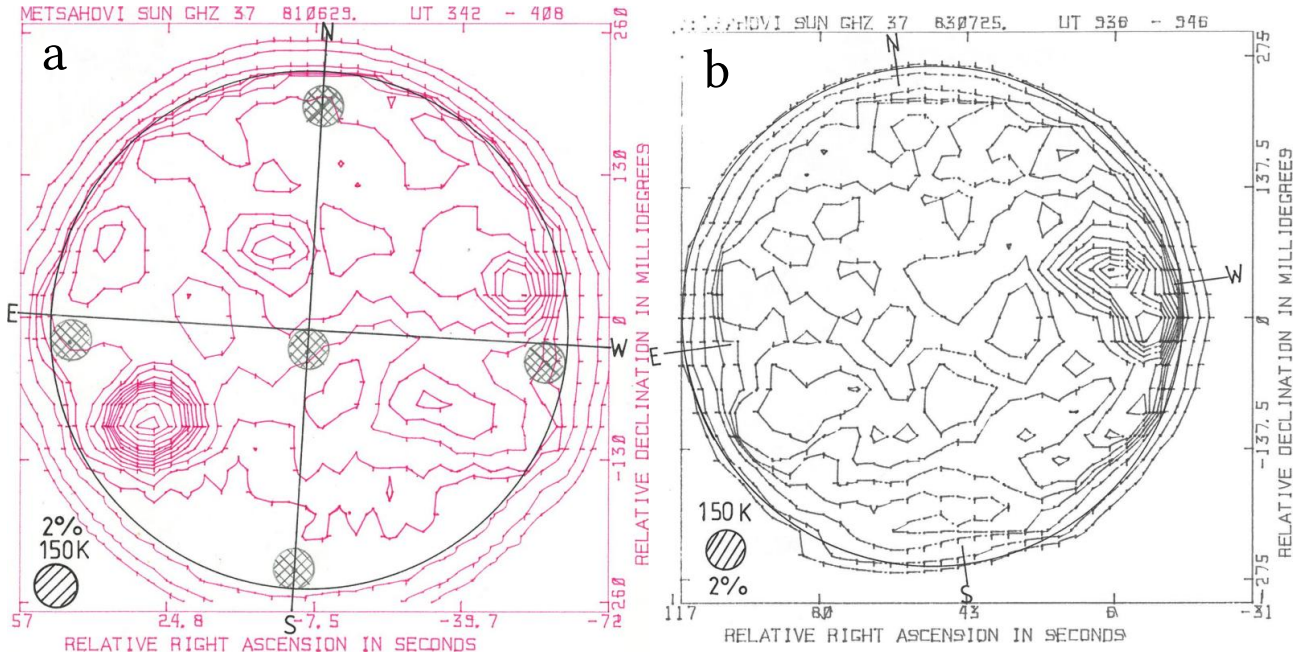

Figure 13: Examples of contours defects. a) The map contains external markings which interfere with contour detection, resulting in broken contours. b) Original maps had problems drawing the contour lines, so that kinks have missing ink.

Figure 14: *An example of a brightening at the rim. Contour directions are denoted with arrows. The contours typically rotate the rim counterclockwise, but when there is a sharp brightening at the rim, the flow is clockwise (marked with 'A').*

2.1 Selecting the best colour channel

Some of the scanned files are greyscales, when some are coloured images. From the coloured images we will notice, that sometimes the maps were drawn with magenta and sometimes with blue or green ink. The scanned images have three channels, red (r), green (g), and blue (b). In order to reduce data size, only one channel is chosen for processing. To aid with the selection, we calculate the variance for each channel as:

$$\mathrm{Var}\,(c_i) = \frac{n * \sum\limits_{(x,y)} c_i^2(x,y) - \left(\sum\limits_{(x,y)} c_i(x,y)\right)^2}{n^2} \qquad i \in \{\mathrm{r},\mathrm{g},\mathrm{b}\}\,. \tag{7}$$

For example, magenta ink has no green component, whereas white background has all of these channels close to maximum illumination. It is then feasible to

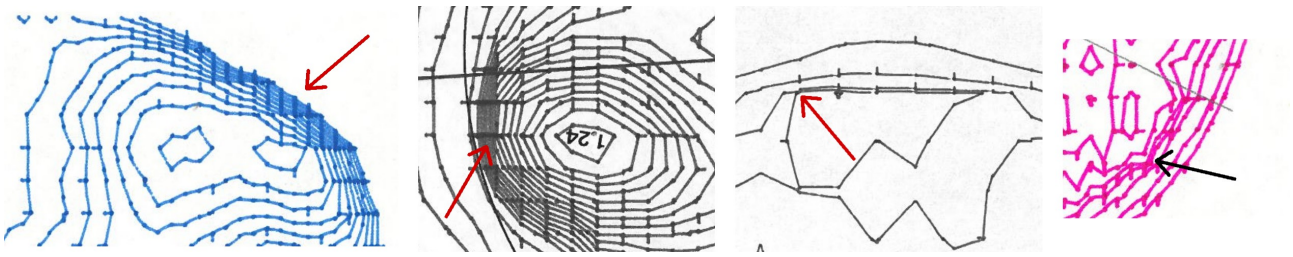

Figure 15: *Examples of overlapping contour lines marked with arrows. These occur in regions where the intensity gradient is exceptionally high, such as near the visible disk terminator. They may also be quantisation or interpolation effects produced when plotting.*

use the green channel for digitisation. For blue ink, we will choose either red or green channel, but even better signal-to-noise ratio would be obtained by taking a weighted linear combination of red and green, which is would then resemble a yellow filter. So far, we have processed the maps with only one color channel. In the future, the maps may be analysed with respect to a linear combination of colours. This is currently not a high priority, since more quality improvement is achievable by other means. Also, if the compass rose is hard to detect, we may later utilise the colour information in layout detection, since the compass rose is always drawn with strong black ink.

2.2 Filtering

The scanned documents have line widths of approximately eight pixels. The algorithms used downstream in the digitising pipeline are more reliable when the map is smooth in short length scales. This can be observed in Fig. 17 by plotting the intensity cross-sections.

This requires filtering out the spatial frequencies corresponsing to short distances. The image may also have large regions with different background brightness resulting from shading during scanning or from degradation of the paper. For this reason, it is useful to include a set of filters. An image with dimensions $x_{\mathrm{dim}} \times y_{\mathrm{dim}}$ can be defined as a function $p(x, y) : \ D \mapsto \mathbb{R}$ where

$$D = [0, x_{\mathrm{dim}} - 1] \times [0, y_{\mathrm{dim}} - 1] \subset \mathbb{Z}^2. \tag{8}$$

It is useful to express p through its Fourier transform, which is defined as:

$$\mathcal{F}\left(p\right)\left(k_{\mathrm{x}}, k_{\mathrm{y}}\right) := \hat{p}(k_{\mathrm{x}}, k_{\mathrm{y}}) = \sum_{(a,b)\in D} e^{-2\pi i\left(\frac{k_{\mathrm{x}} a}{x_{\mathrm{dim}}} + \frac{k_{\mathrm{y}} b}{y_{\mathrm{dim}}}\right)} p(a, b) \qquad (k_{\mathrm{x}}, k_{\mathrm{y}}) \in \mathbb{R}^2, \tag{9}$$

so that there can be defined an inverse Fourier transform:

$$\mathcal{F}^{-1}\left(\hat{p}\right)\left(x, y\right) \ := \ \frac{1}{x_{\mathrm{dim}} y_{\mathrm{dim}}} \sum_{(k_{\mathrm{x}},k_{\mathrm{y}})\in D} e^{2\pi i\left(\frac{x k_{\mathrm{x}}}{x_{\mathrm{dim}}} + \frac{y k_{\mathrm{y}}}{y_{\mathrm{dim}}}\right)} \hat{p}(k_{\mathrm{x}}, k_{\mathrm{y}}) \tag{10}$$

$$= \frac{1}{x_{\text{dim}} y_{\text{dim}}} \sum_{(a,b),(k_\text{x},k_\text{y})\in D} e^{2\pi i\left(\frac{xk_\text{x}-k_\text{x}a}{x_{\text{dim}}}+\frac{yk_\text{y}-k_\text{y}b}{y_{\text{dim}}}\right)} p(a,b) \tag{11}$$

$$= \frac{1}{x_{\text{dim}} y_{\text{dim}}} \sum_{(a,b)\in D} \left(\sum_{k_\text{x}=0,\ldots,x_{\text{dim}}-1} e^{2\pi i \frac{k_\text{x}}{x_{\text{dim}}}(x-a)} \right) \left(\sum_{k_\text{y}=0,\ldots,y_{\text{dim}}-1} e^{2\pi i \frac{k_\text{y}}{x_{\text{dim}}}(y-b)} \right) p(a,b) \tag{12}$$

$$= \sum_{(a,b)\in D} \delta_{a,x}\delta_{b,y} p(a,b) = p(x,y). \tag{13}$$

Here we have used the notation for Kronecker delta, which is defined as:

$$\delta_{a,x} = \begin{cases} 1 & \text{when } a = x \\ 0 & \text{otherwise} \end{cases}. \tag{14}$$

We can then set $P := \mathcal{F}^{-1}(\hat{p}) : \mathbb{R}^2 \mapsto \mathbb{R}$ which is actually defined in real space. P then contains periodic tiles of p:

$$P(x + nx_{\text{dim}}, y + my_{\text{dim}}) = P(x,y) \qquad n,m \in \mathbb{Z}. \tag{15}$$

The image p filtered through $f : D \mapsto \mathbb{R}$ is a convolution:

$$(f * p)(x,y) = \sum_{(a,b)\in D} F(x-a, y-b) * p(a,b), \tag{16}$$

where F is periodic in the sense of Eq. 15. Calculating $(f * p)$ directly would require $O\left(N^2\right)$ operations (with $N := x_{\text{dim}} y_{\text{dim}}$), but less work is required when we are using the Fourier coefficients $\hat{p}$ and $\hat{f}$:

$$(f * p)(x,y) = \frac{1}{N^2} \sum_{(a,b),(k_\text{x},k_\text{y}),(l_\text{x},l_\text{y})\in D} e^{2\pi i\left(\frac{(x-a)k_\text{x}+al_\text{x}}{x_{\text{dim}}}+\frac{(y-b)k_\text{y}+bl_\text{y}}{y_{\text{dim}}}\right)} \hat{f}(k_\text{x},k_\text{y})\hat{p}(l_\text{x},l_\text{y}) \tag{17}$$

$$= \frac{1}{N^2} \sum_{(a,b),(k_\text{x},k_\text{y}),(l_\text{x},l_\text{y})\in D} e^{2\pi i\left(\frac{xk_\text{x}+a(l_\text{x}-k_\text{x})}{x_{\text{dim}}}+\frac{yk_\text{y}+b(l_\text{y}-k_\text{y})}{y_{\text{dim}}}\right)} \hat{f}(k_\text{x},k_\text{y})\hat{p}(l_\text{x},l_\text{y}) \tag{18}$$

$$= \frac{1}{N} \sum_{(k_\text{x},k_\text{y}),(l_\text{x},l_\text{y})\in D} \delta_{k_\text{x},l_\text{x}}\delta_{k_\text{y},l_\text{y}} e^{2\pi i\left(\frac{xk_\text{x}}{x_{\text{dim}}}+\frac{yk_\text{y}}{y_{\text{dim}}}\right)} \hat{f}(k_\text{x},k_\text{y})\hat{p}(l_\text{x},l_\text{y}) \tag{19}$$

$$= \frac{1}{N} \sum_{(k_\text{x},k_\text{y})\in D} e^{2\pi i\left(\frac{xk_\text{x}}{x_{\text{dim}}}+\frac{yk_\text{y}}{y_{\text{dim}}}\right)} \hat{f}(k_\text{x},k_\text{y})\hat{p}(k_\text{x},k_\text{y}) = \mathcal{F}^{-1}\left(\hat{f}\hat{p}\right)(x,y). \tag{20}$$

The convolution is obtained by calculating the Fast Fourier Transform (FFT) of functions $f$and g, and performing the inverse FFT on their product $\hat{f}\hat{p}$. This involves $O\left(y_{\text{dim}} * x_{\text{dim}} \log x_{\text{dim}} + x_{\text{dim}} * y_{\text{dim}} \log y_{\text{dim}}\right) = O\left(N \log N\right)$ operations.

Gaussian band pass filter can be constructed by defining two wavelengths λ_{min} and λ_{max} with their corresponding wavenumbers $h_{\text{min}} := \frac{2\pi}{\lambda_{\text{min}}}$ and $h_{\text{max}} := \frac{2\pi}{\lambda_{\text{max}}}$. The

idea for the filter is that the middle wavenumber $h_{\mathrm{mid}} = \frac{1}{2}(h_{\mathrm{min}} + h_{\mathrm{max}})$ has unity gain, while the key wavenumbers have half gain. Given these two parameters, h_{min} and h_{max}, it is convenient to define a band pass filter, since we know there the cutoff wavelenghs are. Details clearly smaller than $0.5h_{\mathrm{min}}$ or greater than $0.5h_{\mathrm{max}}$ will be filtered out.

In general, the gain is a gaussian function:

$$\hat{f}(k_{\mathrm{x}}, k_{\mathrm{y}}) = e^{-\left(\frac{h(k_{\mathrm{x}},k_{\mathrm{y}})-h_{\mathrm{mid}}}{h_{\mathrm{max}}-h_{\mathrm{min}}}\right)^2 * \ln 16}. \tag{21}$$

Wavenumbers are defined as radians per unit length:

$$\boldsymbol{h}(k_{\mathrm{x}}, k_{\mathrm{y}}) = \left(\frac{2\pi k_{\mathrm{x}}}{x_{\mathrm{dim}} * w}, \frac{2\pi k_{\mathrm{y}}}{y_{\mathrm{dim}} * h}\right) \qquad h(k_{\mathrm{x}}, k_{\mathrm{y}}) = |\boldsymbol{h}(k_{\mathrm{x}}, k_{\mathrm{y}})|. \tag{22}$$

We use filters in two stages of the data flow. First time we filter the scanned images, and once we have obtained an intensity field with integer size discontinuities at contour lines, we want to smooth the map for easier analytics. The smoothing utilises non-isometric rectangular pixels with dimensions $w \times h$, $w \neq h$. In order to construct a low pass filter, we will simply set $h_{\mathrm{min}} := -h_{\mathrm{max}}$, so that the unity gain is for zero wavenumber and the above definitions apply.

2.3 Pen path detection

The dark areas in the image are generally interesting features, such as texts, contours, or lines and circles related to the layout of the map. White areas are open space and may contain noise as well as dust particles, dirt, lossy decompression artefacts, etc. In order to reduce the information content of the map, we can make a list of all the pixel values as $(L_j)_j$: $L_j = \left(L_{\mathrm{x}}^{(j)}, L_{\mathrm{y}}^{(j)}, L_{\mathrm{v}}^{(j)}\right)$, $j = 0, 1, 2, ...$ where $\left(L_{\mathrm{x}}^{(j)}, L_{\mathrm{y}}^{(j)}\right)$ are the pixel coordinates and $L_{\mathrm{v}}^{(j)}$ is the brightness value after monochromatisation and band pass filtering. The list is sorted with ascending brightness values as:

$$i \leq j \mathrel{\widehat{=}} L_{\mathrm{v}}^{(i)} \leq L_{\mathrm{v}}^{(j)}. \tag{23}$$

The first item L_0 is added to a list of reference points, which are defined as a list $(C_k)_k$, $k = 0, 1, 2, ...$ The functional feature for these reference points is that they occupy the area within a certain radius $d/2$. It is thus forbidden to have two reference points closer than d, which should correspond to the line width of a pen track on a scanned image. It is supplied to the code as a parameter *cpoint_dist* (see Appendix B.5). When constructing $(C_k)_k$, we will restrict distances to the already existing reference points:

$$\forall j < k : \left(C_{\mathrm{x}}^{(j)} - C_{\mathrm{y}}^{(k)}\right)^2 + \left(C_{\mathrm{y}}^{(j)} - C_{\mathrm{y}}^{(k)}\right)^2 \geq d^2 \qquad C_{\mathrm{v}}^{(j)} \leq C_{\mathrm{v}}^{(k)}. \tag{24}$$

When selecting items from $(L_j)_j$ and sequantially adding them to $(C_k)_k$, we will do the bookkeeping on a grid of boolean values. The grid is initially ($k = 0$) filled with zeros:

$$B \ : \ (x, y, j) \mapsto \{0, 1\} \, , \ B(x, y, 0) = 0 \tag{25}$$

$$\forall (x, y) \in [0, x_{\max} - 1] \times [0, y_{\max} - 1]. \tag{26}$$

When an item $(x', y', v) = L_j$ from pixel list $(L_i)_i$ is added to the list of reference points $(C_i)_i$ as an item C_k, $k \geq 1$, we will tag all the pixels which are at most at distance d from c:

$$B(x, y, k) = \begin{cases} 1 & \text{when } (x - x')^2 + (y - y')^2 \leq d^2 \\ B(x, y, k - 1) & \text{otherwise} \end{cases} . \tag{27}$$

Proceeding with increasing j for $(x', y', v) = L_j$, we will assign L_j into C_k if $B(x, y, k) = 0$. Then we will also increment k. If $B(x, y, k) = 1$ we will skip that particular pixel and only increment j.

We will stop when a certain fraction of the list L is processed. This fraction is specified in a command line parameter *fraction* (see Appendix B.5). When finished, we have a pattern, where the original pen path is partially covered with reference points, while the reference points are absent in the white regions of the image (Fig. 18a).

The subsequent step is to connect neighbouring reference points into chains, and ideally these chains should follow the pen tracks exactly. The distance between adjacent reference point on a single pen track will vary from d to $2d$. This can be observed in Fig. 18a, where the red circles have gaps. These gaps can be up to one diameter wide, since otherwise the algorithm could assign a new reference point in between.

This simple approach works fine as long as the paths are clearly distinct, but fails when two paths have spacing less than $2d$, based on the central curves of the paths. Then it is no longer trivial, based on reference points only, to determine where the pen track is. We would mix the to lines and have inconsistent contours. The line detection will be more robust when the path is fully covered. To achieve this, we will mark the whole path at once after picking the first point on it. We will see no more gaps between the red circles in (Fig. 18b).

Suppose that the reference point c is marked first. We will measure the filtered brightness at points $(a_j)_{j=1}^N$, which surround c and are equally spaces as in Eq. 28. Now two distinct ink strength maxima can be observed in the cycle with $c_j = c_{j+N} \ \forall \ j \in \mathbb{Z}$:

$$a_j = c + d * e^{2\pi i \frac{j}{n}} \qquad a_{\max 1}, a_{\max 2} \in \{a_1, ..., a_n\} \qquad a_{\max 1} \neq a_{\max 2}. \tag{28}$$

For maxima we need the ink strenght $s : \mathbb{R}^2 \mapsto \mathbb{R}$ to satisfy $\forall j = 1, ..., n : s(a_{\max 1}), s(a_{\max 2}) \geq s(a_{\mathrm{j}})$.

We will produce two distict sets of points $C_1 := \{c_{j-w}, ..., c_j, ..., c_{j+w}\}$ and $C_2 := \{c_{k-w}, ..., c_k, ..., c_{k+w}\}$. For both sets, we will fit a parabola as in Fig. 19. Maximum intensity defines the direction where the path proceeds.

In order to determine an accurate direction which maximises the ink strength at distance d, we will fit a parabolic function in the directional intensity curve. The coefficients of the parabola constitute a vector $\boldsymbol{c} = (c_0, c_1, c_2)$. For a set of points $((x_j, y_j))_{j=1}^n$, we need to find such $\boldsymbol{c}$ that the following target function is maximised:

$$T(\boldsymbol{c}) = \sum_{j=1}^{n} (\Delta y_j)^2 = \sum_{j=1}^{n} \left(c_0 + c_1 x_j + c_2 x_j^2 - y_j\right)^2. \tag{29}$$

Since $T(\boldsymbol{c})$ is continuous and differentiable, the minimum must have zero gradient:

$$\nabla T(\boldsymbol{c}) = \begin{pmatrix} \frac{\partial T}{\partial c_0} \\ \frac{\partial T}{\partial c_1} \\ \frac{\partial T}{\partial c_2} \end{pmatrix} = 2 * \begin{pmatrix} \sum \Delta y_j \\ \sum x_j \Delta y_j \\ \sum x_j^2 \Delta y_j \end{pmatrix} = \mathbf{0}. \tag{30}$$

Moreover, ∇T is also differentiable (and more generally, T is analytic), so we can calculate its Hessian matrix [42]:

$$H_T = \begin{pmatrix} \frac{\partial^2 T}{\partial c_0^2} & \frac{\partial^2 T}{\partial c_0 \partial c_1} & \frac{\partial^2 T}{\partial c_0 \partial c_2} \\ \frac{\partial^2 T}{\partial c_1 \partial c_0} & \frac{\partial^2 T}{\partial c_1^2} & \frac{\partial^2 T}{\partial c_1 \partial c_2} \\ \frac{\partial^2 T}{\partial c_2 \partial c_0} & \frac{\partial^2 T}{\partial c_2 \partial c_1} & \frac{\partial^2 T}{\partial c_2^2} \end{pmatrix} = 2 * \begin{pmatrix} 1 & \sum x_j & \sum x_j^2 \\ \sum x_j & \sum x_j^2 & \sum x_j^3 \\ \sum x_j^2 & \sum x_j^3 & \sum x_j^4 \end{pmatrix}. \tag{31}$$

Since the Hessian in constant, $T(\boldsymbol{c}) = T(\mathbf{0}) + \boldsymbol{c} \cdot \nabla T(\mathbf{0}) + \frac{1}{2} \boldsymbol{c}^T H_T \boldsymbol{c}$ and $\nabla T(\boldsymbol{c}) = \nabla T(\mathbf{0}) + H_T \boldsymbol{c}$. The $\boldsymbol{c}$ that minimizes T has:

$$H_T \boldsymbol{c} = -\nabla T(\mathbf{0}) \tag{32}$$

$$\boldsymbol{c} = -H_T^{-1} \nabla T(\mathbf{0}). \tag{33}$$

Once the coefficients $\boldsymbol{c}$ are determined, the exact location of the maximum can be calculated as $x_{\max} = -\frac{-c_1}{2c_2}$.

This is an example of a quadratic optimization problem, in which the solution is unique and directly obtained by solving a set of linear equations. When the Hessian is not constant, we can often still find at least a local extremum by iterating $\boldsymbol{c}_{k+1} := -H_T^{-1}(\boldsymbol{c}_k) \nabla T(\boldsymbol{c}_k)$. Convergence then requires several steps, and we need to have a suitable starting point, $\boldsymbol{c}_0$, for the iteration. In the special case for H_T being positively definite, meaning that $\boldsymbol{g}^T H_T \boldsymbol{g} > 0 \; \forall \, \boldsymbol{g} \neq \mathbf{0}$, we have convex problem set and a global minimum. [42]

Once the two directions are determined, we can start building up the path from the situation in Fig. 19. The next step is shown in Fig. 20, where the previous direction is taken as a hint. The path most likely advances in the previous direction. In the case of Fig. 20, the decision is simple, but for more complicated cases, there may be multiple pen tracks at the vicinity. In that case, the direction that goes straight is preferred, so that a score function is calculated for each possible maximum. The score is a linear combination of the deviating angle and the maximum ink strenght available for the particular direction option. The parameter *sledge_factor*, used for balancing this decision making, is critical for the success of the digitisation. Among with other related parameters, it is listed in Appendix B.7.

So far we have demonstrated the basic principle of line detection used in the code. The practise is more complicated, and the problem is divided into three stages:

1. Detect tips.
2. Collect suitable middle points of paths and extend the path, into two directions, for as long as possible.
3. Finally fill in the rest of the map with reference points. Set them in pixels that are still available within brightness threshold and not too close to other reference points.

Each of the three stages is a loop which selects pixels in brightness order and generates reference points. With the tip detection, there code first tries to optimize the tip location to stretch out as much as possible. This will help later when detecting gradient indicators. The stretching out involves similar quadratic direction optimization, as in Fig. 20. It will advance in small steps and try to avade the tip body.

2.3.1 Detecting tips

For tip selection, the code proceeds as follows:

1. Select next pixel, $:= p$, in ascending brightness order, if any.
2. Measure *tip_crown_dirs* directions with respect to p, equally spaced at distance *tip_crown_dist* from p, and calculate the amount of brightness minima in the obtained data set.
3. If there is more than one clear brightness minimum, we reject the pixel and proceed from step 1.
4. Optimize the previous direction $d := O(d, p)$ based on the local brightness distribution, similar to Fig. 20.
5. Proceed a small step $\epsilon :=$ *tip_optimize_dist* into direction $-d$. Set $p' := p - \epsilon d$.

6. Measure the brightness value at p'. If thebrightness value is above the general threshold, we have successfully detected a step in p and will continue from step 1.
7. If p' is at a location where we already have a path, reject the tip and proceed from step 1.
8. Otherwise we will set $p := p'$ and continue from step 4.

Once a suitable tip has been found, we will build a neck for it. This involves in choosing a direction of advance for the path and extending a path into that direction for up to *tip_max* steps. We will block the subsequent reference points, so that they will no longer be selected as new reference points. If we run into a line formation which contains two or more possible directions of advance, we will terminate the neck extension and leave the last neck reference open. This allows the path tracker to tag the location later. An example of a successful path and tip interpretation is displayed in Fig. 21.

The two-way path tracking also utilises additional optimization steps for best possible reliability. A lot of fine detail tuning has been done with the line detection, and the work continues as the quality of the maps has to improve. One option in the future would be to train a neural network for interpreting the local pen path environment, and generating the decisions of advance from that network. That will require a contour simulator code for generating a useful training set. Basic neural network principles are presented e.g. in [43].

2.4 Detecting layout

One particular task required in interpreting the maps is to recognize circles which are part of the layout. The big circle drawn at the apparent perimeter of the Sun is harder to detect, since it is accompanied with contour lines which also follow the perimeter. It is therefore not possible to obtain a continous path around the big circle.

A knee is a triplet of reference points, such as $\kappa_j = (l_j, c_j, r_j)$. It is required, that there is a pen path connecting l_j and c_j as well as c_j and r_j, meaning that:

$$I\left(\alpha c_j + (1-\alpha) l_j\right) \geq I_{\mathrm{th}} \quad \forall \alpha \in [0,1]. \tag{34}$$

In practice, this is determined by collecting a set of points $(\alpha_t, I_t)_{t=0}^{10}$ with $I_t = I\left(\alpha_t c_j + (1-\alpha_t) l_j\right)$ and fitting a parabola $I(\alpha) = c_0 + c_1\alpha + c_2\alpha^2$ for the set $(\alpha_t, I_t)_{t=0}^{10}$. Position of the extremum is then $\alpha_{\mathrm{ex}} = -\frac{c_1}{2c_2}$. If there is no extremum (a.k.a $\alpha \notin [0,1]$) or the extremum value is beyond the threshold

$$I_{\mathrm{ex}} = c_0 - \frac{c_1^2}{4c_2} \geq I_{\mathrm{tr}}, \tag{35}$$

the points l_j and c_j are considered connected. Equivalently r_j and $c_j$have to be connected for the knee κ_j to exist. It is also required that $l_j \neq r_j$. There is a maximum leg distance allowed: $|l_j - c_j|, |r_j - c_j| \leq d_{\text{knee}}$ as well as a maximum angle $\alpha : \ |\alpha| < \alpha_{\text{knee}}$. Knees are assigned scores according to

$$s_j = f_{\text{dist}} * |l_j - r_j| + \alpha. \tag{36}$$

Here α represents the knee angle. When the reference points are thought as complex numbers, we can denote:

$$e^{i\alpha} = \frac{l_j - c_j}{|l_j - c_j|} * \frac{|c_j - r_j|}{c_j - r_j}. \tag{37}$$

Here f_{dist} is the parameter ***distance_factor*** (see Appendix B.9), which specifies whether to favour short or straight chain segments. It is one of the critical parameters which require intensive optimisation, since for adjacent paths we always need to choose the right option where to extend the chain.

Short distances and straight angles (arguments of complex numbers) are favoured, and the chains are set to follow straight pen tracks when such are available. Two knees,κ_i and κ_j, which are sharing the same central point ($c_j = c_i$) are said to conflict if they also share some of the leg points, a.k.a if either $r_j = r_i$, $r_j = l_i$, $l_j = r_i$, or $l_j = l_i$. For any set of mutually conflicting knees, the one with best score is chosen to be active.

Next we will construct long chains $(\kappa_j)_{j=1}^n$, for which either

$$r_{j+1} = c_j \quad \text{or} \quad l_{j+1} = c_j \qquad \forall j = 1, ..., n-1. \tag{38}$$

Depending on the stage of the process, chains may or may not be allowed to cross. Contours are definitely not allowed to cross other contours, while the layout features may well cross contours and other layout features. Two knees κ_j and κ_m for $j \neq m$ are crossing, iff the legs rotate around a common centre $c_j = c_m$. We will use c_j as an origin and treat the legs as two-dimensional vectors: $a_1 := c_j - l_j$, $a_2 := r_j - c_j$, $b_1 := c_m - l_m$, and $b_2 = r_m - c_m$.

We need to determine which of the points b_1 and b_2 are on the left, compared to the movement $a_1 \to 0 \to a_2$. We denote $s(b)$ for this purpose:

$$s(b) = \begin{cases} \text{sign}\left(\min\left\{a_1 \times b, a_2 \times b\right\}\right) & \text{when } a_1 \times a_2 \geq 0 \\ \text{sign}\left(\max\left\{a_1 \times b, a_2 \times b\right\}\right) & \text{otherwise} \end{cases}. \tag{39}$$

The knees κ_j and κ_m are then crossing, iff $s(b_1) = s(b_2) = 1$.

The initial stage of chain formation produces relatively short chains, which survive uninterrupted lines but are often broken on gradient indicators, sharp kinks,

and when intersecting a layout feature (Fig. 22). The algorithm is merging these contours into larger ones, in which part in the beginning and end of any chain may be neglected. The parameters for guiding this behaviour are listed in Appendix B.9.

Layout consists of circles and lines. As displayed on Fig. 1a, there is a large box which is always a square to the accuracy of a few pixels. Below the square there is a text "relative right ascension in seconds" and on the right "relative declination in millidegrees". Above the box is a title which mentions Metsähovi, the Sun, date and universal time. These texts and the box are drawn with the XY plotter that also draws the contours. The box has usually five equally spaced tick marks on all four sides. The tick marks on opposite sides are exactly facing each other and define a certain value of right ascension on declination. All the ticks are usually well within the box, so that the short tick lines are clearly distinguishable. However, sometimes the tick line coincides with the perpendicular box boundary. The tick line may even appear slightly outside the box. The digitisation code includes features so that it should handle these special cases. The tick line always begins at the box boundary, or its extension, and points towards the accompanying tip at the opposite side. The numbers associated with the ticks are centre aligned with the tick lines. The tick numbers are also drawn with the XY plotter.

Initially, the digitisation code was written to recognize the map layout by first detecting a box, which consists of four sufficiently long lines. The required box line length is one of the tunable parameters, and the code should be able to parse even different maps by tuning this parameter along with other parameters.

First all detected lines are sorted according to their length. Starting with the longest lines, we try with every combination. Suppose there are n lines with sufficient length, we then have $\binom{n}{4}$ possible combinations to try. Having selected the four lines as a box candidate, they are first sorted so that the first and the third line are parallel. We calculate the 6 possible angles between these four candidates, and select the lines forming the smallest absolute angle to be the opposite sides a and c. This implicitly requires that also the second and fourth lines are parallel, and the candidates are now ordered as a, b, c, d. We then calculate the angles of the box candidates and require that $a/b = a/d = c/b = c/d = 90^\circ$ and that $a/c = b/d = 0^\circ$. The allowed tolerance in this is a required parameter. If the angles are not within tolerance, the box candidate is neglected and we take another from the $\binom{n}{4}$ possible combinations.

Next step is to calculate the four intersection from perpendicular lines. These will form the four corners of the box. We are treating the lines as infinite, so that if the layout detection produces a broken line, or if the line is extended by a spurious marking on the map, the digitisation code is not mislead. Having

calculated the corners A, B, C, D, we re-define the box sides as $a := B - A$, $b = C - B$, $c := D - C$, and $d := A - D$. We ensure that the rotation $ABCD$ is always anticlockwise when seen on the bitmap, and objects are swapped when necessary. Bitmap pixels are generally coordinated as x increasing from left to right and y increasing from top to bottom. This differs from the usual convention in plots, in which x increases from bottom to top. In the digitisation codes, there are functions for making coordinate transforms between *image* and *map* coordinates. *image* coordinates are the pixels (origin at left top corner, first pixel), while *map* coordinates have an origin at the left bottom corner of the box area, which has maximum value of relative right ascension and minimum value of relative declination. Both coordinate systems have unit one pixel. The *map* usually appears aligned with the *image*, but there is always a small angle offset, since paper sheets can not be scanned straight. The code is able to detect arbitrary *map* orientation, since it begins by detecting the rectangular box.

At this point, the map is still an arbitrary tetragon, but it can be averaged into a rectangle. A set of orthonormal directions is generated as:

$$\hat{a} := \frac{a + ib - c - id}{|a + ib - c - id|} \qquad \hat{b} := -i\hat{a} \qquad \hat{c} := -\hat{a} \qquad \hat{d} := i\hat{a}. \tag{40}$$

We can refine the first corner A in order to produce more redundant estimate, A' based on the averaged dimension of the rectangle and the neighbouring line segments:

$$A' = \frac{A+B}{2} - \left[\left(\frac{A+B}{2} - \frac{A+D}{2}\right) \cdot \hat{a}\right] \hat{a} \tag{41}$$

$$= \frac{A+B}{2} - \left[\frac{B-D}{2} \cdot \hat{a}\right] \hat{a}. \tag{42}$$

The other corners are treated equivalently:

$$B' = \frac{B+C}{2} - \left[\frac{C-A}{2} \cdot \hat{b}\right] \hat{b} \tag{43}$$

$$C' = \frac{C+D}{2} - \left[\frac{D-B}{2} \cdot \hat{c}\right] \hat{c} \tag{44}$$

$$D' = \frac{D+A}{2} - \left[\frac{A-C}{2} \cdot \hat{d}\right] \hat{d}. \tag{45}$$

We will neglect the box suggestion if it produces an aspect ratio that deviates too much from unity. The box detection loop stops when a suitable box is found.

2.5 The compass rose

The origin still remains to be decided, and there are several ways to select it. It should be the corner with highest right ascension and lowest declination. The Sun's rotational north axis points to a fixed point relative to distant stars. It is

also sufficiently fixed in the equatorial coordinate system, and has values RA = 286.13° and dec = +63.87°. The Sun's north axis thus makes an angle 26.13° to Earth's north axis. The rotational north axis is usually marked in the Metsähovi maps, and it can make at most 26° angle with the declination axis of the map. If we are able to recognize the north, we can assign the origin in the *map* coordinates.

In the original scanned plots, the apparent disk of the Sun is drawn manually with a pair of compasses. Usually it intersects with some of the outermost contours, but sometimes the drawn circle is completely outside all the contours. This diversity is observed in Fig. 23. Accompanying the compass circle are four direction indicators for (N)orth, (S)outh, (W)est and (E)ast, which are drawn with ruler and marked with tilted capital letters. The letters are drawn with a set of stencils and have a fixed size. The length of the direction indicators wary, and they may be inside or outside the circle, or intersect it. The digitisation code takes several parameters which restrict how the compass rose is interpreted. A same set of parameters works for most of the maps, but some maps may require parameter tuning from the command line.

There is a known sensitivity issue when the compass circle is very close to a uniform and smooth contour. 19 of such maps were failed with the common set of parameters, but were fixed when the circle detection algorithm used more strict parameters. This process will possibly be more robust with some improvements in the circle and arc detection code. Curretly it is not able to drop obsolete and conflicting reference points, adopted at early steps, even if the arc detection data structure has since then grown towards much better circular fitting and collected new reference points. For more information, see 4.1.

A similar algorithm, as for the box detection, is used for selecting the compass rose direction indicators. All interpreted lines are given a weight parameter based on their length and how much they stick out of the compass circle. Most favourable lines are selected first for the next step, where we compare the alignment of the indicators. The algorithm stops when a suitable orthogonal direction indicator combination is found. There is a caveat in this approach, since sometimes the maps contain short lines that are formed when contour gradient indicators are overlapping at the perimeter regions with dense contours (Fig. 23). If we end up with n possible lines, there are $\binom{n}{4}$ combinations to try. This was not a problem with the box, since the box requires very long lines which are rare. A solution suggested in 4.1 would index the line candidates better.

Sometimes the direction indicators are drawn for the full diameter of the compass circle, so that their intersection marks the centre of the visible solar disk. In this case, we have only two more lines which should be perpendicular within tolerance. There are then two ways to define the centre, and the compass circle centre

should when coincide with the intersection. A larger tolerance is required for single direction indicators, since a misalignment may arise from the line detection. Letter markings may be interpreted as parts of those lines. Since the long sides of letter 'N', at the tip of a direction indicator, are only a width of half-character out of alignment with the direction indicator. This results in interpretation of tilted lines. It is also hard to detect the tip of the lines accurately.

Some maps have a missing compass circle. The map may be unfinished with no direction indicators added, or a close-up in which the whole Sun is not visible. The close-ups have usually two crossed lines indicating the rotation axis. These lines need not cross if the close-up does not include the centre of the Sun.

2.6 Character recognition or the compass rose

At least one of the direction indicators should contain some of the letters 'N', 'W', 'S', 'E'. If more than one is present, they should be consistent with the rotation order of directions on the sky. For this reason, a tilted rectangular area of the map is cut at the locations where the letters should appear. The cut sections are then analyzed with character recognition. We collect the already recognized reference points from within these sections.

The principle of character recognition in this work follows the typical data flow presented e.g. in [44], [45], and [46]. A similar data flow would be needed already for the contour detection. With a simple research engine survey from the Internet, it appears that most of the character recognition implementations rely on first extracting the characters from the image, and then comparing them to the template characters, which might be parametrized in some form [47], [48], [49], and [50]. The approach presented here will try to find suitable offset and scaling, in order to match two point sets. A metric between two point sets constructed, and the character template which produces least distance to the image section will be selected. The approach shares the basic idea of 1-nearest neighbours algorithm with fixed radius [51], where the points are indexed on a grid for efficiency. Documents considering piecewise quadratic optimisation and the set metric used here were not found in a simple survey, and here the codes are written from scratch without further references.

In order to increase robustness, we also include the midway points in cases where two reference points are connected in terms of brightness threshold (see Eq. 35 for definition). So of the reference points c_i and c_j belong to the same section and are connected, then also $\frac{c_i+c_j}{2}$ will be added to the template. The code does not yet prevent adding midway points arbitrarily close to each other, but an example algorithm is already written for eliminating overlapping reference points.

The inclusion of midway points results in more sharp character figures and did

improve the reliability during developement. Using the filtered pixel values instead of reference points would be another approach, but that would require more computation. The sections should have sufficient height along the direction indicator, since the direction indicator may be lengthened due to line detection flaws. Two different versions of these sections are made. One is excluding the direction indicator and the frame line, as including it might increase a possibility for a false match. The other version includes the direction indicator, since the detected line may contain segments of a character (Fig. 24). An improvement would be to always exclude the frame lines if the timestamp of the map requires the axis to be tilted, so that the letters would not be intersecting with the box boundary.

Originally four templates were extracted from the map, one for each letter. The templates were manually cleaned so that they do not contain anything else but the letters. We compare these eight cut-off sections with each of these four templates, so that 32 optimized comparisons are required.

2.7 Matching characters with templates

The set $\mathcal{T}$ contains the template points, while the set $\mathcal{M}$ contains the reference points from map sections. We apply an isotropic scaling by factor e.g. $\alpha \in [0.85, 1.15]$ and translation by an arbitrary vector in $d \in \mathbb{C}$ (or $d \in \mathbb{R}^2$). We will find such α and d that the target function $Q(\alpha, d)$ is minimized. We first need a setup $\mathcal{P} \subset \mathcal{M}$, which contains the map points which are close to some template point, by a distance at most D:

$$\mathcal{P} := \left\{ m \in \mathcal{M} \mid \exists t \in \mathcal{T} \quad \text{such that} \quad |t - m|^2 \leq D^2 \right\}. \tag{46}$$

If $\mathcal{P} = \emptyset$, we neglect the match, otherwise we define the template function $Q : (\mathcal{R}, \mathbb{C}) \mapsto \mathcal{R}_+$ to be evaluated as

$$Q(\alpha, d) = \frac{\sum\limits_{p \in \mathcal{P}} \min\limits_{t \in \mathcal{T}} |(\alpha p + d) - t|^2}{|\mathcal{P}|} + \frac{\sum\limits_{t \in \mathcal{T}} \max \left\{ D^2, \min\limits_{p \in \mathcal{P}} |(\alpha p + d) - t|^2 \right\}}{|\mathcal{T}|}. \tag{47}$$

The characher recognition is essentially an optimization problem. We need to find the best values for α and d, which will minimise $Q(\alpha, d)$. The output of the matching algorithm is the template function evaluated for the best scaling and translation:

$$Q_{\min} := \min_{\alpha, d} \{ Q(\alpha, d) \}. \tag{48}$$

For neglected maps we set $Q_{\min} := 2D^2$, which is the largest value the template function may possibly have. As a rationale, $Q(\alpha, d)$ is the sum of two directional average distances between sets of points and reflects the quality of a match between the template and the section. $Q(\alpha, d)$ is unaffected by features on the map that are not part of the character. Including more points in the map section

and in the template generally improves quality and reliability, since there is less random effects when creating the sections and templates. The idea to include the midpoints significantly improved the reliability.

It is important to calculate average distances, since the number of points should not be an optimization criterion. Various strategies exist for finding optimal α and d. In this implementation, we first set $\alpha := 1$ and pick a fixed central point from the template, $t_c \in \mathcal{T}$. For initiating the optimization iteration the point closest to the centre of gravity of $\mathcal{T}$ is chosen. Then we try for each $m_c \in \mathcal{M}$ and set $d := t_c - m_c$, so that a particular point in the cut-off section is mapped to this central point in the template. We then calculate $\mathcal{P}$ as in Eq. 46. We will find a closest neighbour $t_p \in \mathcal{T}$ for each point in $p \in P$. When there are multiple options, we have freedom to choose:

$$\forall p \in \mathcal{P} : \text{ set } t_p := t \text{ such that } t \in \mathcal{T} \text{ and } |(\alpha p + d) - t|^2 \text{ is minimal.} \tag{49}$$

Similarly, we will choose a closest neighbour p_t for each $p \in \mathcal{P}$ as:

$$\forall t \in \mathcal{T} : \text{ set } p_t := p \text{ such that } p \in \mathcal{P} \text{ and } |(\alpha p + d) - t|^2 \text{ is minimal.} \tag{50}$$

We then set a local template function which coincides at specific scaling and translation: $Q(1, d) = Q'(1, d)$:

$$Q'(\alpha, d) = \frac{\sum\limits_{p \in \mathcal{P}} |(\alpha p + d) - t_p|^2}{|\mathcal{P}|} + \frac{\sum\limits_{t \in \mathcal{T}} |(\alpha p_t + d) - t|^2}{|\mathcal{T}|}. \tag{51}$$

It is a quadratic optimization problem to minimize Q', since $H_{Q'}$ does not depend on α or d.

$$\nabla Q'(\alpha, d_x, d_y) = \left(\frac{\partial Q'}{\partial \alpha}, \frac{\partial Q'}{\partial d_x}, \frac{\partial Q'}{\partial d_y} \right)^T H_{Q'} = \begin{pmatrix} \frac{\partial^2 Q'}{\partial \alpha^2} & \frac{\partial^2 Q'}{\partial \alpha \partial d_x} & \frac{\partial^2 Q'}{\partial \alpha \partial d_y} \\ \frac{\partial^2 Q'}{\partial \partial d_x \partial \alpha} & \frac{\partial^2 Q'}{\partial d_x^2} & \frac{\partial^2 Q'}{\partial d_x \partial d_y} \\ \frac{\partial^2 Q'}{\partial \partial d_y \partial \alpha} & \frac{\partial^2 Q'}{\partial d_x \partial d_y} & \frac{\partial^2 Q'}{\partial d_y^2} \end{pmatrix}. \tag{52}$$

The optimal is achieved by setting:

$$(\alpha, d_x, d_y) - = H_{Q'}^{-1} \nabla Q'(\alpha, d_x, d_y). \tag{53}$$

We will then recalculate $\mathcal{P}$ (Eq. 46) and form new pairs for the points (Eq. 49 and 50). If the pairs have not changed, we have reached at least a local minimum in Q and are satisfied. This usually happens after five to ten steps. We can then set $Q'_{m_c} := Q'(\alpha, d)$ and repeat for a different point $m_c \in \mathcal{M}$. The minimum so obtained is:

$$Q'_{\min} := \min \left\{ Q'_{m_c} \mid m_c \in M \right\}. \tag{54}$$

This is the calculated match value for a particular map section and template, but it is not necessarily the true optimum $Q_{\min}$, which would be the global minimum in $Q(\alpha, d)$. It would an interesting study to find out when $Q_{\min} = Q'_{\min}$.

There is a threshold value for $Q_{\min}$, otherwise we don't consider a match. The maximum distance D is also an important parameter. An example of a good match is in (Fig. 25).

Having calculated all the required 32 match values, we calculate a sum $\sum {Q'}_{\min}$ for each four possible orientations for the symbols 'N', 'W', 'S', 'E'. We will primarily prefer matches which are obtained with excluding the direction indicator points which might be misinterpreted as parts of characters. We want to recognize as many direction characters as possible. If there is still question which direction to choose, the orientation with lowest sum wins. Denote with $Q_{i,a,b}$ for the $Q_{\min}$value obtained when matching template $i \in \{\mathrm{N}, \mathrm{W}, \mathrm{S}, \mathrm{E}\}$ with a map section $a \in \{1, 2, 3, 4\}$ and mode $b \in \{0, 1\}$. If and only if $b = 1$, the previously detected line features are included in the map section. We use the mode $b = 0$ when such a match is available:

$$Q_{i,a} := \begin{cases} Q_{i,a,0} & \text{when } Q_{i,a,0} < 2D^2 \\ Q_{i,a,1} & \text{otherwise} \end{cases}, \tag{55}$$

and calculate the sums:

$$S_0 = S_{\mathrm{N},0} + S_{\mathrm{W},1} + S_{\mathrm{S},2} + S_{\mathrm{E},3} \tag{56}$$
$$S_1 = S_{\mathrm{N},1} + S_{\mathrm{W},2} + S_{\mathrm{S},3} + S_{\mathrm{E},0} \tag{57}$$
$$S_2 = S_{\mathrm{N},2} + S_{\mathrm{W},3} + S_{\mathrm{S},0} + S_{\mathrm{E},1} \tag{58}$$
$$S_3 = S_{\mathrm{N},3} + S_{\mathrm{W},0} + S_{\mathrm{S},1} + S_{\mathrm{E},2}. \tag{59}$$

The sum $S_i \in \{S_0, S_1, S_2, S_3\}$ with lowest value is selected and north symbol is when found at map i. The rest of the directions are when set accordingly. Reference points that were part of the selected matches will be excluded from further processing, so that they would not be misinterpreted as contours. The contours will, however, be allowed to cross such forbidden reference points, but not allowed to travel from one forbidden reference point to another.

2.8 Coordinate system

If the directions of Sun north axis was successfully detected, we also know where the other labels are, since the Sun's rotational axis is not significantly tilted relative to the ecliptic plane. Otherwise we can fall back to the fact that the texts "relative right ascension in seconds" and "relative declination in millidegrees" are always oriented at the same distance from the box. On the remaining two sides there are no labels.

2.8.1 Text drawn by the plotter

Text labels occur at known locations, so we can cut those labels from the image and and feed them to a text recognition program. One particular program, *ocrad*, which is part the GNU project [52], did not produce reliable results, propably

because the typeface used is obscure. Based on documentation available in [52], the program utilises an ad-hoc algorithm for each character. Since the program is oriented to reading printed text, the approach might not be optimal for this application. We thus follow the footsteps of the earlier example for compass rose directions (2.6). An alternative approach would have been to start developing *ocrad* in parallel with the *sunmap* project, but this might constitute too much overhead.

In order to collect enough samples to construct a set of templates for a character, set, code was developed for separating words from a line of text, as well as individual characters from a word. For other applications, there is also similar code for separating individual lines from a page. In order to increase reliability, several samples of a particular character were stacked by first applying appropriate transformation for each image, and then calculating the average brightness values by pixel.

The character recognition here is similar to the detection of direction indicators, except that here we use half-pixel resolution instead of the reference point diameter, which we used earlier. This is because the xy-plotter drawn characters are much smaller and have finer detail. We do not need to exclude points from the map section, since there should be no external features on top of XY-plotter drawn text. These conditions suggest a binary search method for finding the optimum, since analytically defined parts of Q function constitute very small subsets of the domain.

Trouble was encountered due to some maps containing alternative typeface. Actually some characters were identical, but others clearly bigger. For the first style the maps had e.g. "METSÄHOVI SUN 37 GHZ", while the equivalent for the second style would be "METSÄHOVI SUN GHZ 37". This offered an easy way to tell which parameters to use for character recognition. The question is more about how to separate characters than how to match them to templates, so the same template set can be used for both typefaces. When cutting templates and map sections for character recognition, the bitmaps are always slightly rotated and aligned with the box.

There are also stencil-drawn labels near the antenna beam size indicator. The exact location of these vary, and after trying to hard-code them, there were always some maps which offered surprises. As a solution, the code now extracts large sections from the map, around the beam size indicator, and interprets the sections as pages of text. The algorithm needed for separating lines from pages is identical to the ones for separating words or characters, except that we need to rotate the map by 90°. Only the texts which were correcly interpreted will be forbidden for contour detection, and all other features are treated as contours. The beam

size indicator is completely forbidden, since the people who have published the maps have manually chosen a good location for the beam indicator, and it will not intersect any contours.

2.8.2 Tick label redundancy

Once the tick markers are interpreted, we need to establish the coordinates for both axes. The algorithm for establishing one axis requires a set of numbered pairs $\{(x_j, z_j),\ j = 1, ..., n\}$ where x_j is the distance from the origin along the particular axis. y_j is the value interpreted from the cut-off section of text. Ideally, there should be linear mapping $y_j = a + bx_j$, so that the coefficients arise from linear regression model. We need to find such $a, b \in \mathbb{R}$ that minimize the target function:

$$T(a,b) = \sum_{j=1}^{n} (a + bx_j - y_j)^2 . \tag{60}$$

This occurs at the point (a, b) where $\nabla T(a, b) = 0$. The partial differentials are

$$\frac{\partial T}{\partial a} = 2\sum_{j=1}^{n} (a + bx_j - y_j) = 0 \quad \Rightarrow \quad a = \frac{\sum_{j=1}^{n} y_j}{n} - b\frac{\sum_{j=1}^{n} x_j}{n} \quad \text{and} \tag{61}$$

$$\frac{\partial T}{\partial b} = 2\sum_{j=1}^{n} (a + bx_j - y_j)\, x_j = 0 \quad \Rightarrow \quad a = \frac{\sum_{j=1}^{n} x_j y_j - b\sum_{j=1}^{n} x_j^2}{\sum_{j=1}^{n} x_j}. \tag{62}$$

We can use the notation $\langle \cdot \rangle$ for average and write:

$$\begin{cases} a &= \langle y \rangle - b\langle x \rangle \\ b &= \frac{\langle xy \rangle - b\langle x^2 \rangle}{\langle x \rangle} \end{cases} . \tag{63}$$

When we solve the linear system in Eq. 63, we will obtain an expression for the slope b. It follows:

$$\langle y \rangle \langle x \rangle - b\langle x \rangle^2 = \langle xy \rangle - b\langle x^2 \rangle \quad \Rightarrow \quad b = \frac{\langle xy \rangle - \langle x \rangle \langle y \rangle}{\langle x^2 \rangle - \langle x \rangle^2} = \frac{\mathrm{Cov}(x, y)}{\mathrm{Var}(x)}. \tag{64}$$

The target function can then be evaluated as

$$T(a,b) = \sum_{j=1}^{n} \left[b\left(x_j - \langle x \rangle\right) - \left(y_j - \langle y \rangle\right) \right]^2 . \tag{65}$$

The target function is useful, since its value describes how well the tick markers satisfy the linear dependency. We can obtain a more convenient way to calculate

the error in the dependency as

$$\frac{T(a,b)}{n} = b^2\left(\langle x^2\rangle - \langle x\rangle^2\right) - 2b\left(\langle xy\rangle - \langle x\rangle\langle y\rangle\right) + \left(\langle y^2\rangle - \langle y\rangle^2\right) \quad (66)$$

$$= \frac{[\mathrm{Cov}(x,y)]^2}{\mathrm{Var}(x)} - 2\frac{[\mathrm{Cov}(x,y)]^2}{\mathrm{Var}(x)} + \mathrm{Var}(y) = \mathrm{Var}(y)\left(1-\gamma^2\right). \quad (67)$$

Here $\gamma \in [-1:+1]$ is the Pearson correlation coefficient, which is ± 1 when there is no error in the linear model:

$$\gamma := \frac{\mathrm{Cov}(x,y)}{\mathrm{Var}(x)\,\mathrm{Var}(x)}. \quad (68)$$

It is efficient to calculate the required sums $\sum x_i$, $\sum y_i$, $\sum x_i^2$, $\sum x_i y_i$, and $\sum y_i^2$ in a loop, and then calculate the variances as:

$$\mathrm{Var}(x) = \langle x^2\rangle - \langle x\rangle^2 \qquad \mathrm{Cov}(xy) = \langle xy\rangle - \langle x\rangle\langle y\rangle. \quad (69)$$

2.8.3 Priorities for fallbacks coordinates

When some tick numbers are misinterpreted, the scaling of the map will be wrong. In order to have a more robust algorithm, we will try to find a subset which produces the best correlation. However, if we simply optimize the correlation, we will end up with any subset which contains only two pairs $\{(x_j,y_j),(x_k,y_k)\}$ with $x_j \neq y_j$ and $y_j \neq y_k$. The approach here is to find such a pair which agrees with maximal number of other pairs. We will run the test for all j and k with $1 \leq j < k \leq n$. We will skip the test if $|x_j - x_k| < \epsilon$ or $|y_j - y_k| < \epsilon$. Otherwise we will calculate a linear model:

$$b_{j,k} := \frac{y_j - y_k}{x_j - x_k} \qquad a_{j,k} := y_j - x_j b_{j_k}. \quad (70)$$

Then we form a subset $Q_{j,k}$ which contains the only the pairs which agree with $a_{j,k}$ and$b_{j,k}$:

$$Q_{j,k} := \left\{(x_m,y_m) \mid (a_{j,k} + b_{j,k}x_m - y_m)^2 \leq \Delta^2\right\}. \quad (71)$$

We will also calculate the Pearson correlation for each subset:

$$\gamma_{j,k} := \frac{\mathrm{Cov}_{Q_{j,k}}(x,y)}{\mathrm{Var}_{Q_{j,k}}(x)}, \quad (72)$$

and select the set with most pairs. That will have largest cardinality $|Q_{j,k}|$. If there is more than one such set, we will select the one with maximal $\gamma_{j,k}^2$. The final linear model is then calculated from the chosen subset:

$$b_{\boldsymbol{RA/dec}} := \frac{\mathrm{Cov}_{Q_{j,k}}(x,y)}{\mathrm{Var}_{Q_{k,j}}(x)} \qquad a_{\mathrm{RA/dec}} := \langle y\rangle_{Q_{j,k}} - b\langle x\rangle_{Q_{j,k}}. \quad (73)$$

So far we have established the scales for relative declination and relative right ascension, based on the tick labels below and left of the axes. The obtained scale

should be reasonable, and for full solar maps it is ≈ 1.1 arc seconds per pixel in the used scanning resolution. The scaling is ignored if it is below 0.1 or above 2.0 arcseconds per pixel. In the future, we can enforce this scaling policy already at the pair loop 2.8.2 and avoid obtaining unfeasible scaling from the pair loop and then ignoring it later.

There are also other ways to define coordinates. The filename of the map includes the frequency, date, and time. The same information is also in the title of the map, written using the XY plotter. If there is a conflict between them, we will assume the filename is correct, since it is already confirmed in Metsähovi. From the timestamp we can calculate Sun's apparent position and size. The apparent radius varies between 0.262° and 0.271° degrees due to the eccentricity of the Earth's orbit.

Typically the map has a circle drawn with a pair of compasses, which represent the visual size of the Sun. The centre of the circle should coincide with the origin of the tick-based coordinates, and the scales should agree with the known size of the visible solar disk based on astronomical calculations. For most scanned maps, it appears that the relative right ascension is cosine-corrected, both for the tick label based coordinates and for the visual appearance on the scanned map:

$$\Delta\mathrm{RA}_{\mathrm{cc}} = \cos(\mathrm{dec}) * \Delta\mathrm{RA}. \tag{74}$$

This means that both axes have equal scaling ($b_{\mathrm{RA}} = b_{\mathrm{dec}}$), and the Sun looks circular on the scanned map. Some of the scanned maps did not have cosine correction, so that visible disk is oblate (Fig. 26a). setting *cosine_corrected_img=0* and *cosine_corrected_map=0* will then fix the problem (Fig. 26b), Appendix B.3). The eccentricity of the Sun is ignored.

If there is intolerable conflict in the obtained scales, such that $\left|\frac{b_{\mathrm{RA}}-b_{\mathrm{dec}}}{b_{\mathrm{RA}}+b_{\mathrm{dec}}}\right| > \delta$, we will invalidate the other axis and set:

$$\text{either} \quad b_{\mathrm{RA}} := b_{\mathrm{dec}} \quad \text{or} \quad b_{\mathrm{dec}} := b_{\mathrm{RA}}, \tag{75}$$

based on which scale is more reasonable or agrees better with the marked Sun radius. The origin of the corrected axis will be set based on the cross, or centre of the compass rose. If no such reference is available, we will try fitting for the empirical solar disk that can be constructed from the contours. An axis may also be invalidated if its origin deviates too much from the marked Sun's centre.

2.9 Antenna beam size indicator and related labels

One important feature is the antenna beam size indicator, which contains either parallel lines, or a number of arcminutes. It is reasonable to think that when there are parallel lines, they should represent the scanline spacing, which is the difference in declination between adjacent sweeps. In the proximity of the beam

size indicator circle, there is either a temperature number (typically 140 K), or percentage (typically 2%), or both. When both are visible, their ratio is obviously the Quiet Sun Level (here 7000 K). The Quiet Sun Level varies slightly. The meaning of these two numbers is the intensity difference between adjacent contours. These numbers are drawn with stencils, and their exact position varies. Large sections of texts need to be cut for analysis, and when a reasonable parsing result is obtained, the associated characters are marked forbidden, so that they are not later interpreted as contours.

2.10 Identifying gradient direction indicators

One critical feature in the maps is the gradient direction indicator. Unless the pen path detection and chain forming and merging works perfectly, the gradient indicators may be falsely interpreted as kinks in contours. It is crucial to have as reliable gradient indicators as possible, since the sign of contours will be depending on them. Local maxima are covered with contours that have gradient indicators pointing outwards, while local minima have gradient indicators pointing inward. This is equivalent to height contours in geographic maps.

Fig. 27 demonstrates various challenges related to tip and gradient indicator detection. The blue circles are the first reference points obtained from the tip detection stage described in 2.3.1. Sometimes the tip cannot be detected, since there are other features at the vicinity. The most typical case is when multiple tips are overlapping.

When taking a first glance at almost any of these Metsähovi maps, we will notice that the gradient indicators are organized in a line pattern where the interval of the lines is equal. The gradient indicators clearly have a periodic distribution. If we could measure this periodicity, mistakenly interpreted gradient indicators which do not fit to this periodic pattern can be ignored. This will significatly increase the reliability of the digitisation process.

For this reason a statistical analysis is made of the location of the contours. The XY plotter appears to draw the possible gradient indicators at periodic intervals. We collect the features from paths that are most propably gradient direction indicators, based on several tunable parameters. A probable gradient indicator has a line tip and path emerging in two directions. In order to increase reliablity, the paths must diverge in different directions, and there is a minimum angle between them. There may not be contours around the tip head, only at the base. If gradient indicators point left or right, the y coordinate will have a periodic distribution. Likewise, if it points up or down, the x coordinate is periodic.

Having collected a set of probable gradient indicators, we try to detect the periodicity in their distribution. Large histograms $(x_j)_j$ and $(y_j)_j$, with $j = 0, ..., 2047$,

are collected. They will contain the measured density: x_j is the number of vertical gradient indicators with $x \in [j, j+1)$, and similar logic applies for y for horizontal indicators. Ideally both histograms would resemble a section of a Dirac comb $f(x) = \sum_{m \in \mathbb{Z}} \delta(x - m)$. The offsets, o_x and o_y, are must be obtained from statistics, such that:

$$h_{\mathrm{x}}(\alpha) = \begin{cases} N_x\left(\frac{\alpha - o_x}{\lambda_x}\right) & \text{when } \frac{\alpha - o_x}{\lambda_x} \in \mathbb{Z} \\ 0 & \text{otherwise} \end{cases}, \tag{76}$$

and

$$h_{\mathrm{y}}(\alpha) = \begin{cases} N_y\left(\frac{\alpha - o_y}{\lambda_y}\right) & \text{when } \frac{\alpha - o_y}{\lambda_y} \in \mathbb{Z} \\ 0 & \text{otherwise} \end{cases}. \tag{77}$$

Where $N_x, N_y \ : \ \mathbb{Z} \mapsto \mathbb{Z}$ are integer functions which count how many gradient direction indicators appear at certain period number. Elsewhere there are no gradient indicators.

The Fourier transform of the Dirac comb is also a Dirac comb. For a discrete equivalent, we may define a function Δ with peak interval m and window n.

$$\Delta(x) = \begin{cases} 1 & \text{when } x = jm \text{ for some } j = 0, 1, 2, ... \qquad x = 0, 1, ..., N-1 \\ 0 & \text{otherwise} \end{cases} \tag{78}$$

We will also require that n is divisible by m and set $p := n/m$. The Fourier series of Eq. 78 become:

$$\hat{\Delta}(k) = \sum_{x=0,...,N-1} \Delta(x) \exp\left(-2\pi \mathrm{i} \frac{xk}{n}\right) = \sum_{j=0,1,...,p-1} \exp\left(-2\pi \mathrm{i} \frac{jk}{p}\right) \tag{79}$$

Terms in Eq. 79 add constructively when $k = 0, p, 2p, ..., (m-1)p$. Otherwise they will cancel out, and thus $\hat{\Delta}(k)$ is also a series of peaks:

$$\hat{\Delta}(k) = \begin{cases} p & \text{when } k = jp \text{ for some } j = 0, 1, 2, ..., (m-1)p \\ 0 & \text{otherwise} \end{cases} \tag{80}$$

When we calculate the Fourier series of the histograms in Eq. 76 and 77, we will also observe a series of harmonic peaks, although noise arises since not all the peaks have equal height, and there is a also a contribution from misinterpreted markings. Indicators slightly out of phase will result in reduced amplitude in higher frequencies, so that a typical result looks like Fig. 28. The more harmonic

frequencies are seen, the more regular is the histogram. Each peak has an amplitude and phase. If the spacing between gradient indicators is exacly $\frac{2048}{n}$, with n integer, we would observe only one channel in any peak. The phase would tell the offset, so that vertical gradient indicators might appear at location (x, y) when

$$\exp\left[\frac{2\pi \mathrm{i} n_x}{2048} x * \frac{\hat{x}(n_x)}{|\hat{x}(n_x)|}\right] = 1 \quad \text{with } y \text{ arbitrary,} \tag{81}$$

while horisontal gradient indicators might appear when

$$\exp\left[\frac{2\pi \mathrm{i} n_y}{2048} y * \frac{\hat{y}(n_y)}{|\hat{y}(n_y)|}\right] = 1 \quad \text{with } x \text{ arbitrary.} \tag{82}$$

Unfortunately, there is no hope for expecting n_x and n_y to be integers. In the Fourier series of an ill-tuned Dirac comb, each peak will contain several channels, so that their amplitude is relative to $\frac{\sin(\alpha)}{\alpha + 2\pi\Delta n}$, $\Delta n = 0, \pm 1, \pm 2, ...$ (Appendix C). The peak has a maximum where the amplitude changes sign. Both tails are decaying as $1/n$. In order to disinguish individual peaks in a noisy environment, we will first calculate the average power σ_0 of the whole spectrum. Next, only the channels with power less than 3σ will be included when calculating a new average power, σ_1. The process will be repeated until $(\sigma_i)_i$ appears to converge. The channels with power less that $3\sigma_\infty$ will be considered noise, while those above will be considered peaks-segments. Adjacent peak segments are collected to individual peaks, which should then have left and right decaying tails with opposite amplitudes.

When we plot the amplitudes on a complex plane:

$$\left\{\hat{x}\,(n) \mid n = n_{\min}, ..., n_{\max}\right\}, \tag{83}$$

we can fit a line which will be parallel to the amplitude of the peak, having unit direction $\hat{e}$. This adds redundancy compared to measuring only one channel. We can then replace the complex amplitudes with real amplitudes:

$$a_i := \hat{e} \cdot \hat{x}\,(n_{\min} + i) \qquad i = 0, ..., n_{\max} - n_{\min}. \tag{84}$$

The real amplitude a and wavenumber $\frac{2n\pi i}{2048}$ of the whole peak is obtained by fitting a decaying function to the peak. Optimal $a, n \in \mathbb{R}$ are such that minimize the target function T:

$$T(a, \varphi) = \sum_{i=0}^{n_{\max} - n_{\min}} \left(a * \frac{\sin(2n\pi)}{n_{\min} + i - n} - a_i\right)^2, \tag{85}$$

where the possible division by zero is avoided by treating $\frac{\sin 0}{0} = \operatorname{sinc} 0 = 1$. The real amplitude a is converted back to complex amplitude $p \in \mathbb{C}$by multiplying with the fitted phasor: $\boldsymbol{a} := \hat{e} * a$.

Having now collected all the peaks $(n_j, \boldsymbol{a}_j)_j$ in ascending frequency, we need to find harmonics. We seek for each $i = 1, ..., m$, that how many peaks have frequency n_j, which is approximately an integer multiple of n_i. A certain threshold is needed. Suppose $n_1, n_2, ..., n_m$ are the harmonic frequencies detected. Ideally we could state that $n_k = k * n_1$, but in a noisy environment this is not true. We will thus require that:

$$\left| \frac{n_{k+1} - \frac{k+1}{k} * n_k}{\frac{1}{k} * n_k} \right| \leq 0.05 \tag{86}$$

during each step when finding progressive harmonics n_k. The final frequency is the one with most harmonics and, by definition, also the lowest harmonic.

When the wavenumber of a peak is close to a harmonic multiple of the histogram length L, e.g. $k \approx \frac{2\pi n}{L}$, the decaying tails are not detectable above the noise level. When the peak has no tail, it can only be assigned integer frequencies. To cope with this limitation, the periodicity detection is run with 10 different histogram sizes, and we should get a very similar base frequency n_j, period p_j, phase φ_j, and offset o_j every time. Offset and period relate to frequency and phase as:

$$p_j := \frac{N}{n_j} \qquad o_j := p_j * \frac{\varphi_j}{2\pi}. \tag{87}$$

Of these different results, we choose the one which agrees with maximal amount of other results. Then we will calculate the average frequency and offset, based on these results which agree. When averaging offsets, we need to operate in the complex plane since an offset $o_j \approx 0$ is equivalent to $o_j \approx p_j$:

$$\langle p \rangle = \frac{\sum p_j}{10} \qquad e^{2\pi i \frac{\langle o \rangle}{\langle p \rangle}} = \left| \sum e^{2\pi i * \frac{o_j}{p_j}} \right|. \tag{88}$$

To assess the degree of periodicity in the gradient indicator distribution, we can calculate the signal-to-noise ratio as:

$$r = \frac{\sum\limits_j |\boldsymbol{a}_j|^2}{\sum\limits_{i=0}^{\lceil L/2 \rceil} |\hat{x}(i)|^2}. \tag{89}$$

Ideally we should have r close to unity, but in practise even $r = 0.02$ is sufficient.

2.11 Detecting contours

Now that all the external features are identified from the map, we will tag them as forbidden points. We will run the chaining algorithm again, but in stage we do not allow chains to intersect, since contours never do that. The contour is allowed to cross a layout feature, such as a line, but it is not allowed to travel along it.

This approach does not necessarily break a contour which crosses a layout feature, but often still do when the layout feature makes a low angle with the contour. An example of this second stage of chain joining is shown in Fig. 29.

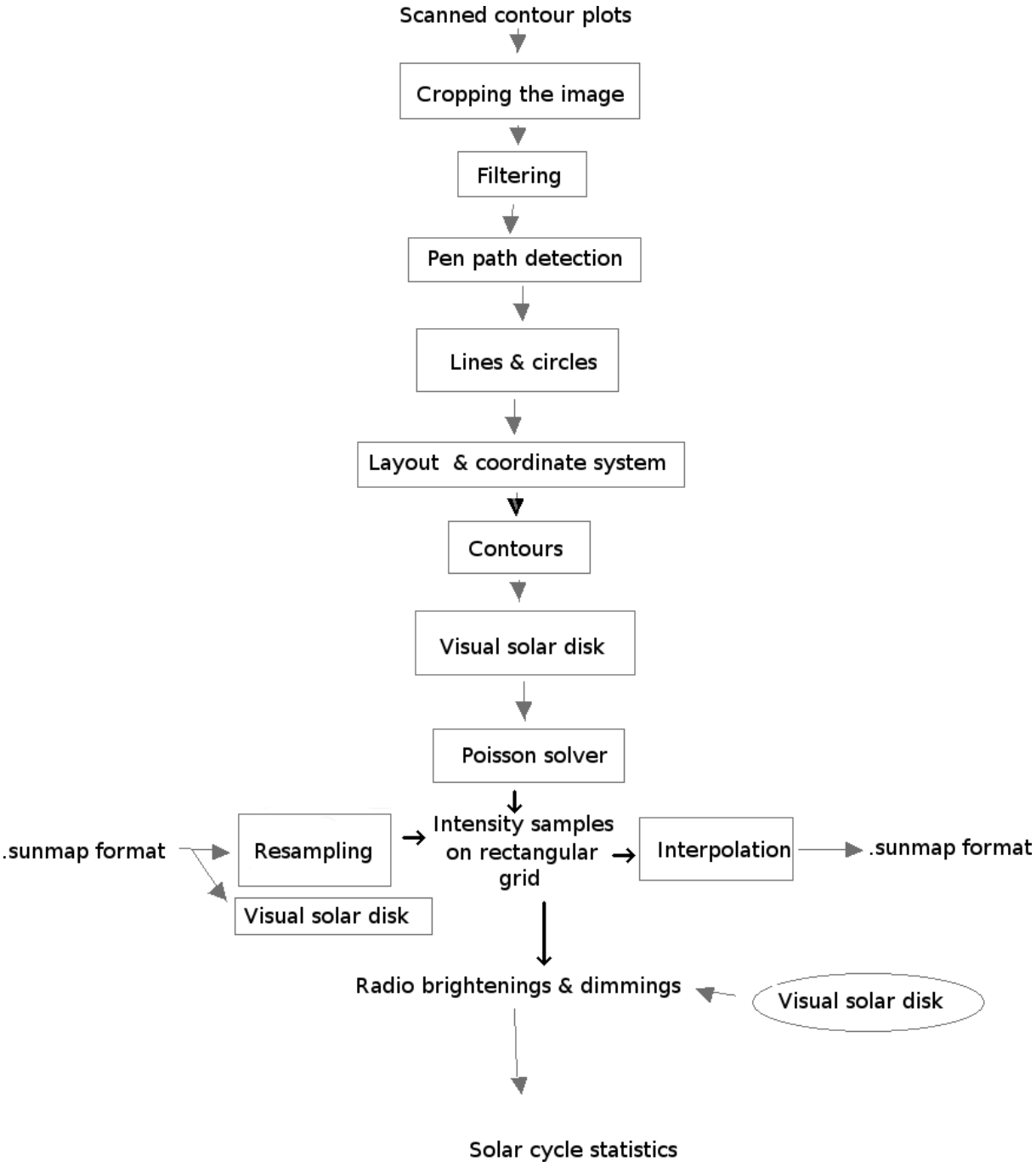

Figure 16: *Flow diagram for* sunmap.

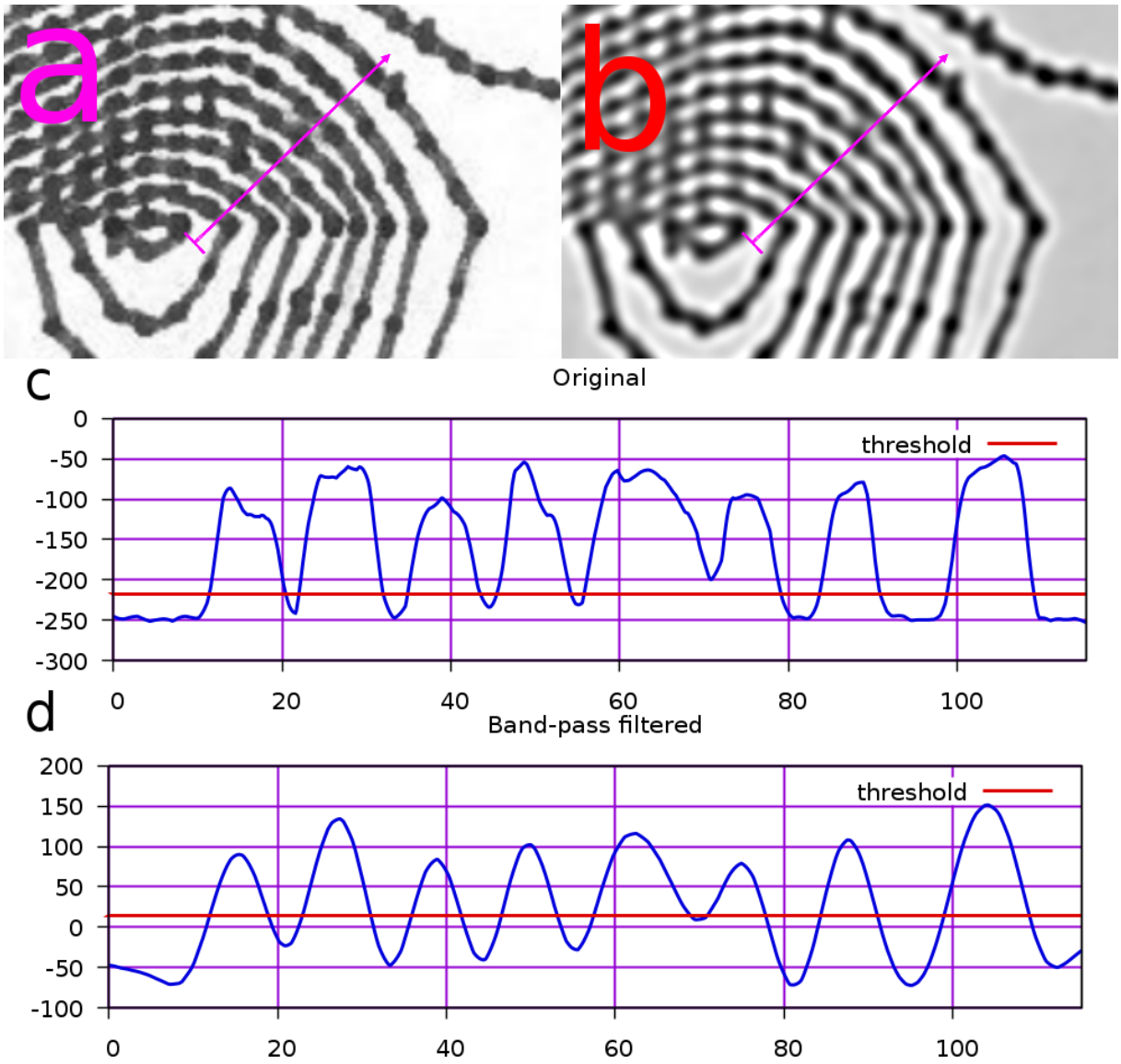

Figure 17: *Brightness profiles measured along the red arrows. a) Original contour image. b) Band pass filtered contour image. c) For unfiltered images, the local maxima are not necessarily in the middle region in lines. d) After suitable band pass filtering, the maxima are nicely identified. It is sufficient to compare single pixel values when seeking for extremum.*

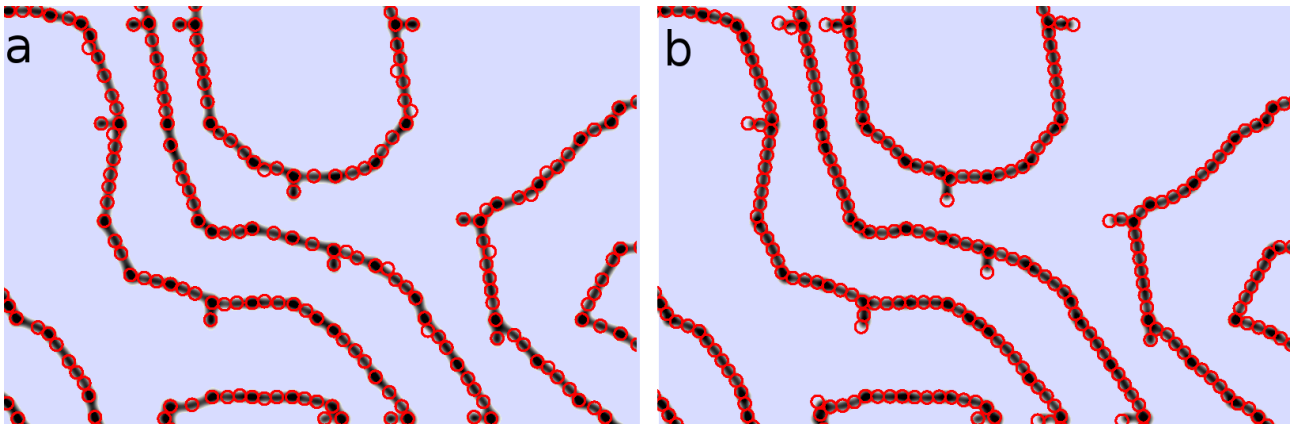

Figure 18: *a) Partially covered pen track when the reference points are assigned in brightness order. b) Emphasis on detecting tips and connected paths.*

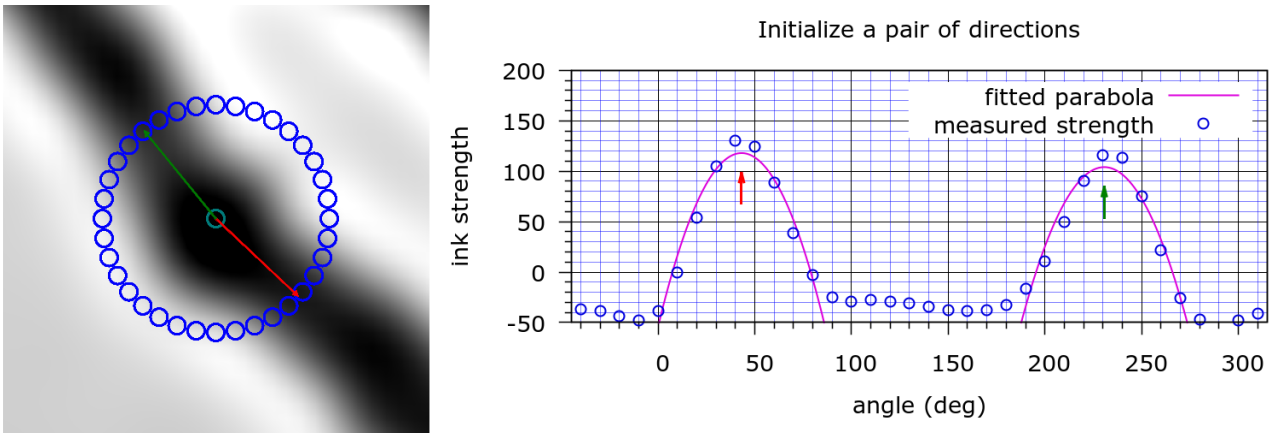

Figure 19: *Choosing a pair of directions for initiating a line from a single reference point.*

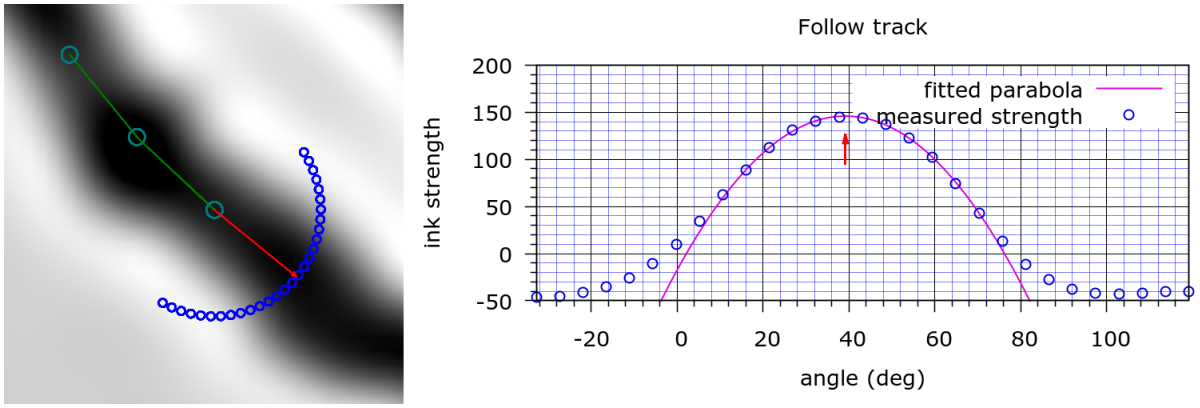

Figure 20: *Continuing a path. a) Measurements are taken at equal distance in the previous direction of advance. b) Maximum ink strength and the corresponding angle is obtained from parabola fitting.*

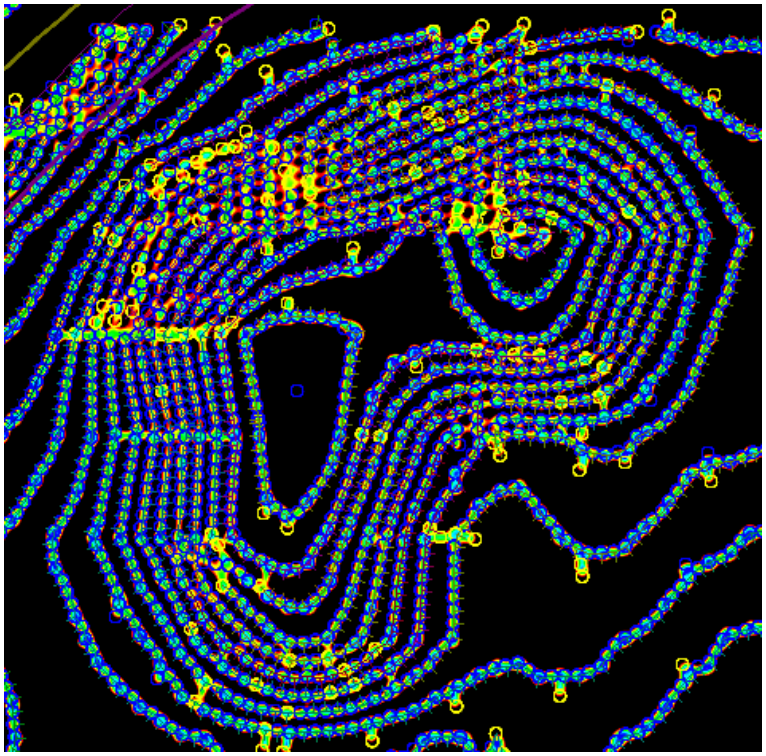

Figure 21: *Tips are identified first to occupy some area, so that the pen path tracking is more reliable.*

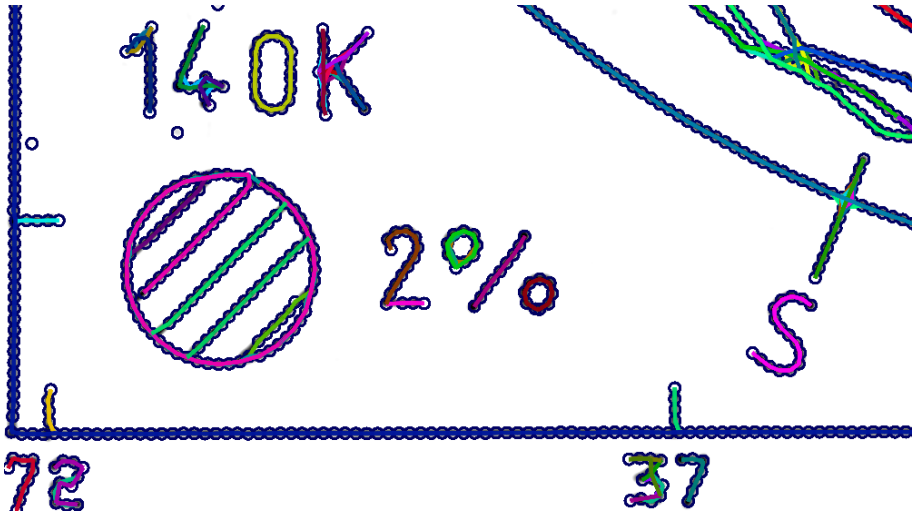

Figure 22: *Detecting layout features by connecting reference points by knees and arranging them into chains. The chains are allowed to cross, as layout features commonly do that.*

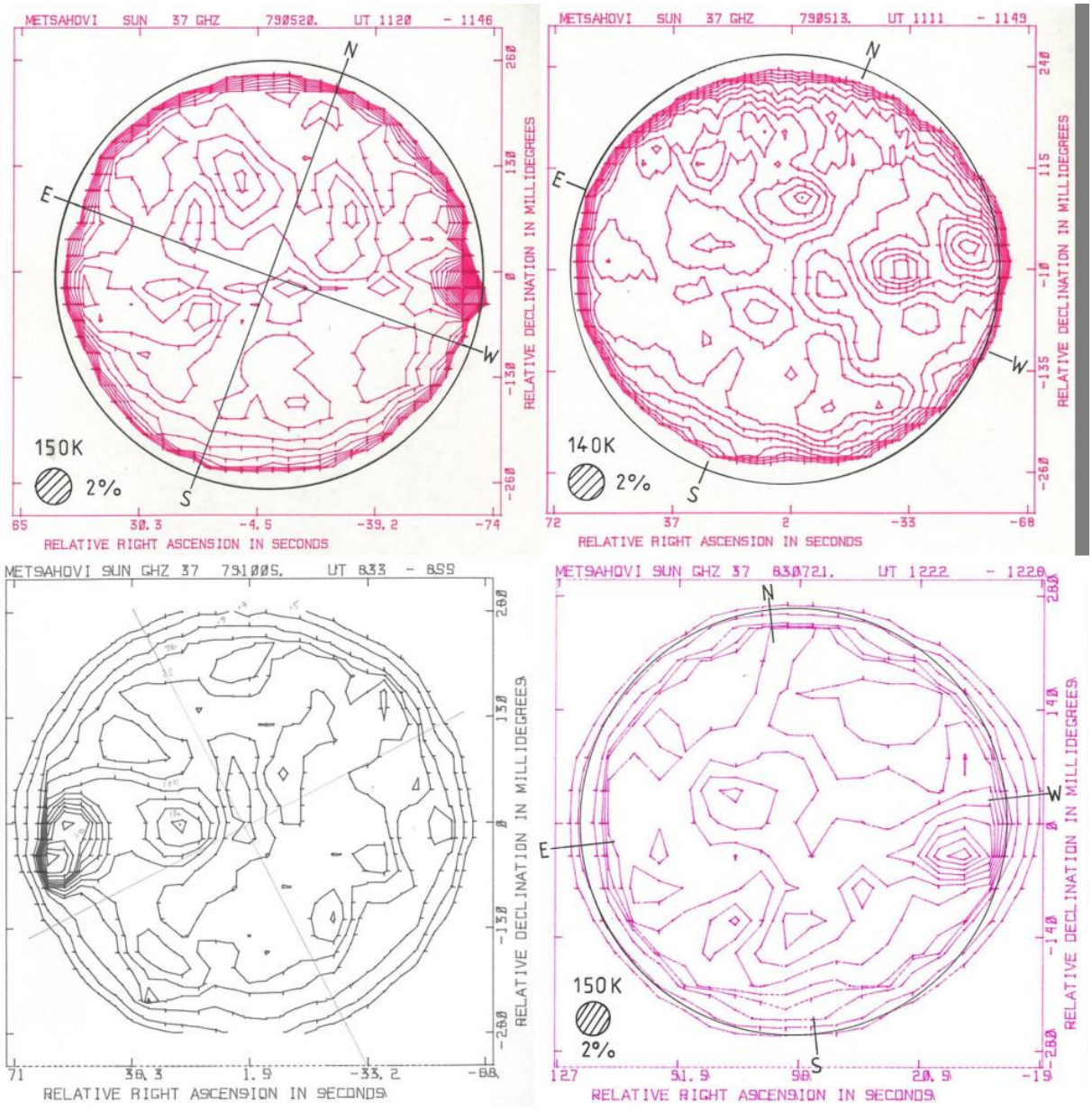

Figure 23: *Some example maps with differences in compass rose.*

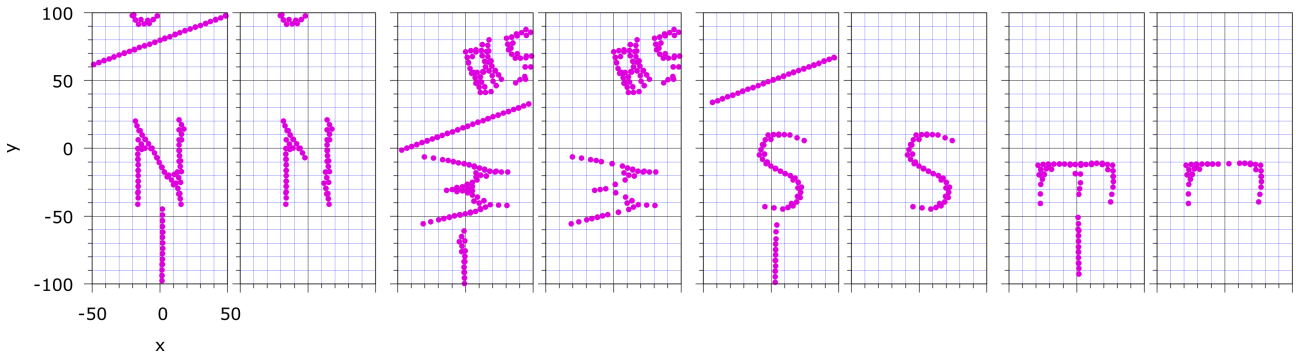

Figure 24: *Each direction letter is extracted with two different modes. Mode $b = 1$ is to include all points, including lines that are propably part of the layout. In the second mode $b = 0$ they are omitted. In the mode $b = 0$, we may be missing points that should be part of a letter.*

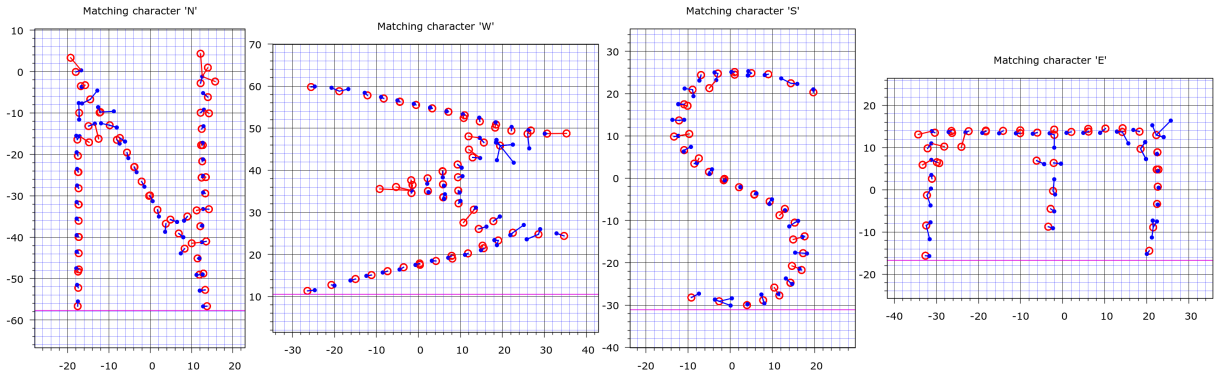

Figure 25: *Matching indicator characters to corresponding templates*

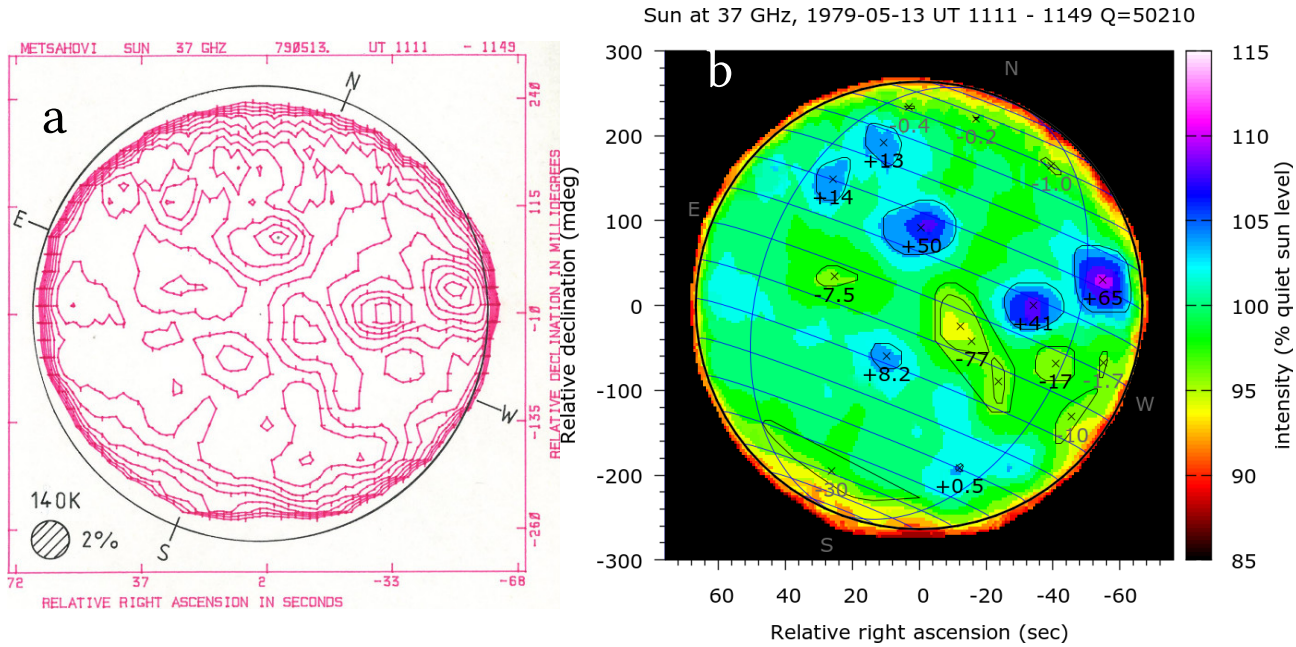

Figure 26: a) Originally flattened solar disk due to lack of cosine correction. b) Map processed with appropriate cosine correction settings.

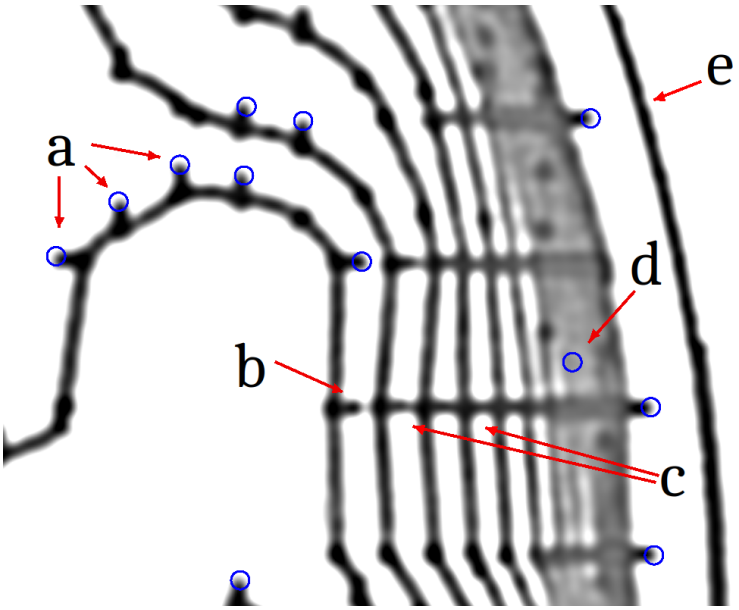

Figure 27: *Detecting tips of gradient indicators. a) Some correctly detected tips. b) A tip went undetected since it was too close to another line. c) Overlapping tips become a unified line. d) A bogus tip arises when filtered ink strength at a contour-covered area falls close to the threshold level. e) Layout feature: arc of a circle marking the expected radius of the visible solar disk.*

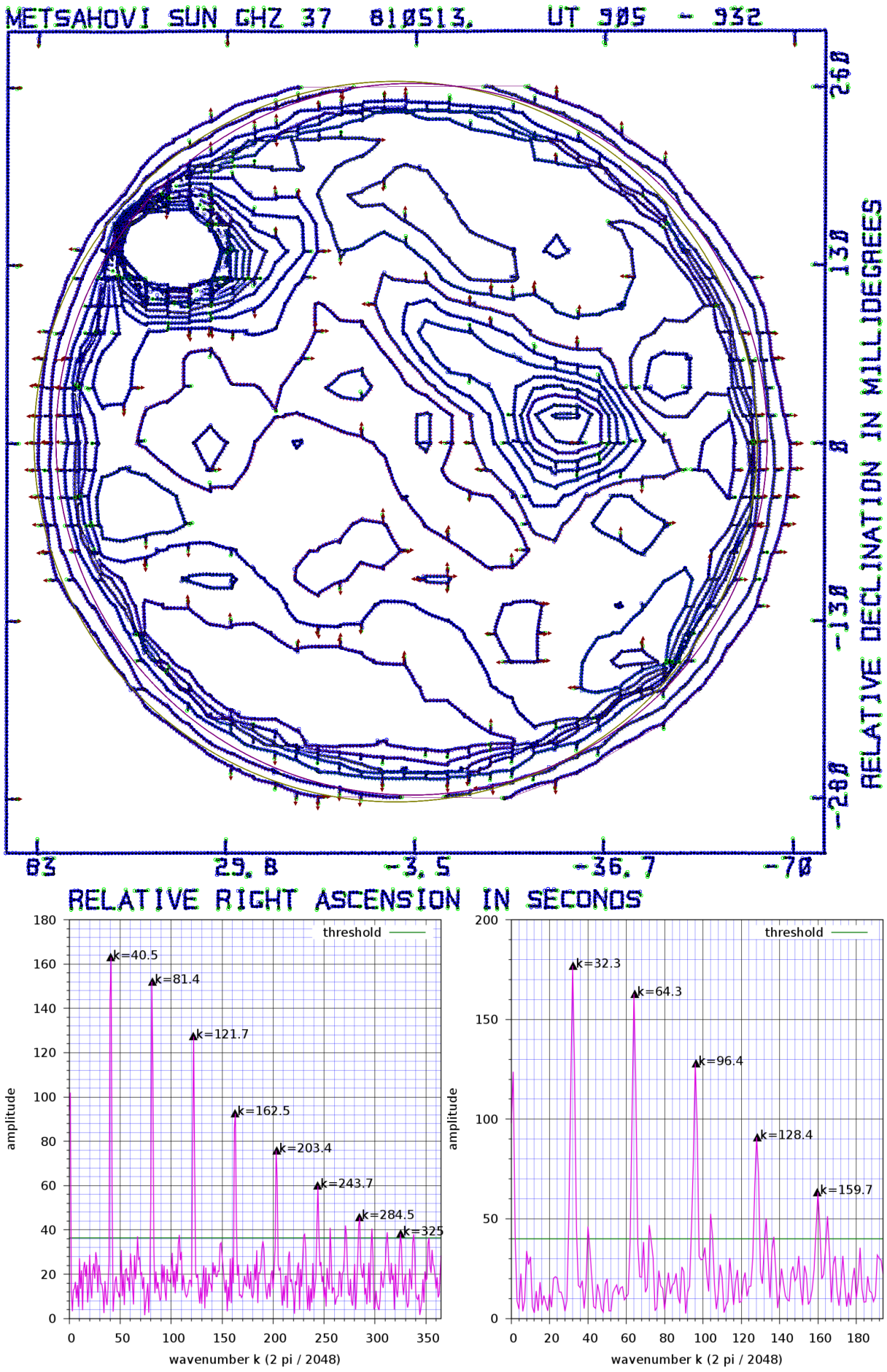

Figure 28: *a) An example map with gradient indicators drawn as red arrows. b) Spectrum of the distribution of x coordinates of the indicators pointing up or down. Signal-to-noise ratio is* 0.13. *3) Spectrum of the distribution of y coordinate of those indicators which point left or right, resulting in signal-to-noise ratio* 0.11.

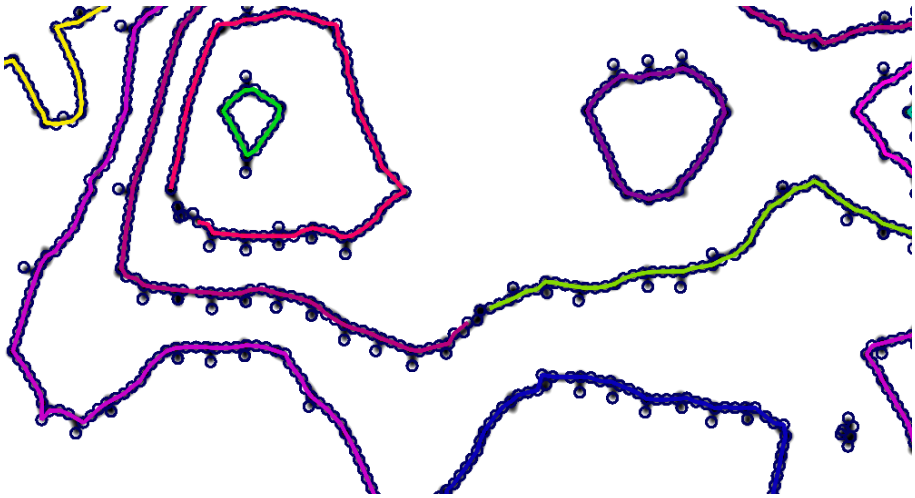

Figure 29: *Second stage of chain joining when they are not allowed to intersect. Individual contour segments are marked with random colors.*

2.12 The Poisson solver

The fundamental idea with contours is that the value of a scalar field has a transition across a line. We aim to convert contour data into a scalar field, and it is feasible that such scalar field has a constant integer value between contours and unity discontinuities whenever there is a contour line. We will later transform the integers on this map into appropriate signal levels which represent physical intensities on the solar disk. This is a linear transformation if the contour levels are equally spaced. When we need a smooth scalar field, such as when finding bright and dim regions, we will also have to use a low pass filter.

Here we have a physical analogy to electric charges and an electric potential set by those charges. We obviously need two dimensional space, but it is equivalent to having a symmetry over the z axis. The system thus does not depend on the z-coordinate. In order to represent contours, we will need a thin sheet of capasitor plate, which is a double line in two dimensional space. We then require unity potential difference between the plates. When we relaxate the potential everywhere else, and all contours are complete, there will be no gradients in the potential. This is because the potential $f(x, y)$ follows Laplace's equation:

$$\nabla^2 f(x, y) = 0, \tag{90}$$

when there are no contours present. In other words, the potential is *harmonic*. If a harmonic region is completely encompassed by a contour, the potential will be constant in the perimeter, and thus $f(x, y)$ will be constant within the region.

The contribution from contours will be treated by adding an appropriate spatial dependency into the Laplace's equation, thus making it a Poisson's equation:

$$\nabla^2 f(x, y) = g(x, y). \tag{91}$$

In a continuum, $g(x, y)$ can not be defined when (x, y) belongs to a contour. However, in order to obtain a numeric solution, we already need to have a discretised system, and here we use a rectangular grid for this purpose. The discretisation allows us to get past the definition problem in Eq. 91. We can use the famous 5-point stencil for this purpose [53]. When i and j are integers representing a grid point, Eq. 91 is satisfied when

$$g_{i,j} = \frac{1}{4}\left(-4f_{i,j} + f_{i+1,j} + f_{i-1,j} + f_{i,j+1} + f_{i,j-1}\right). \tag{92}$$

2.12.1 Poisson solver implementation

The map uses right ascension for x-axis and declination for y-axis. Before proceeding the algorithm with the contours, any point $(\mathrm{RA}, \mathrm{dec})$ in the map has to be linearly converted into contour coordinates, as follows:

$$x := (N-1)\frac{\mathrm{RA}_{\max} - \mathrm{RA}}{\mathrm{RA}_{\max} - \mathrm{RA}_{\min}} \qquad y := (N-1)\frac{\mathrm{dec} - \mathrm{dec}_{\min}}{\mathrm{dec}_{\max} - \mathrm{dec}_{\min}}. \tag{93}$$

For the purpose of this work, a contour c can be defined as a finite sequence of points in $X = [0, N-1]^2$, which is a closed rectangular subset of $\mathbb{R}^2$:

$$c = (c_i)_{i=0}^{m} = (x_i, y_i)_{i=0}^{m}. \tag{94}$$

The intuitive contour line is formed by connecting the subsequent points with line segments. For a closed contour, $c_0 = c_m$. The contour is equivalent to a corresponding finite sequence of directed line segments, where the direction is a complex number at the unit circle:

$$c \mathrel{\widehat{=}} \bar{c} \qquad \bar{c} := (l_i, d_i)_{i=1}^{m}, \tag{95}$$

where we denote $|\bar{c}| = m$ for the cardinality of a sequence. The directions need to be normalised as:

$$l_i := \{\alpha c_i + (1-\alpha)c_{i-1} \ : \ \alpha \in [0,1]\} \text{ and } d_i := \frac{c_i - c_{i-i}}{|c_i - c_{i-i}|} \in \partial\mathcal{B}_2(\mathbf{0}, 1). \tag{96}$$

Every recognised point on the map is part of one contour at maximum. We can collect all the contour information C into a single set of directed line segments:

$$C := \bigcup_{\bar{c}} \{(l_i, d_i) \ : \ i = 1, ..., |\bar{c}|\}. \tag{97}$$

Suppose first that all the contours are intact. We will construct a rectangular grid on top of the contour map. There are $N \times N$ points in the grid, so that the coordinates will be $(x_j, y_j) = (m, n)$ when $j = nN + m$. There will be 2, 3, or 4 pairs of points associated with each grid point j. Before these pairs are listed in the set A_j, we need to keep within the grid by intersecting with B_j:

$$A_j = \begin{Bmatrix} ((m,n),(m+1,n)), & ((m,n),(m-1,n)), \\ ((m,n),(m,n+1)), & ((m,n),(m,n-1)) \end{Bmatrix} \cap B_j, \tag{98}$$

where

$$B_j = \{((m,n),(m',n')) \in A_j \mid m', n' \in \{0, 1, ..., N-1\}\}. \tag{99}$$

Each grid point is associated with a scalar potential value, which is denoted as $v_j = v_{n,m}$. Each potential value v_j depends on the neighbouring cells within the map, where every pair in B_j constitutes a dependency $((m,n),(m',n')) = (v_j, v_k)$ with $k = j' = n'N + m'$ The dependency has a direction:

$$d_{j,k} := (m', n') - (m, n) \tag{100}$$

and a line segment

$$l_{j,k} = \{\alpha\,(m', n') + (1-\alpha)\,(m, n) \ : \ \alpha \in [0,1]\}. \tag{101}$$

The dependency will partition the contour set C into three subsets $L_{j,k}$, $R_{j,k}$, and $F_{j,k}$, based on whether the line segments point left, right, or front, compared to the

dependency direction. We will rotate $d_{j,k}$ by 90° by multiplying with imaginary unit i. The scalar product $\cdot$ has its usual meaning:

$$i * (x, y) = (-y, x) \qquad (x_1, y_1) \cdot (x_2, y_2) = x_1 x_2 + y_1 y_2. \tag{102}$$

The partition is then:

$$\begin{aligned} L_{j,k} &:= \{(l, d) \in C \ : \ d \cdot (i d_{j,k}) > 0\} && (103)\\ R_{j,k} &:= \{(l, d) \in C \ : \ d \cdot (i d_{j,k}) < 0\} && (104)\\ F_{j,k} &:= \{(l, d) \in C \ : \ d \cdot (i d_{j,k}) = 0\}. && (105) \end{aligned}$$

Intersection of any two sets in a partition is empty. We will further partition C depending on whether the line segments intersect with the dependency:

$$\begin{aligned} L_{j,k,0} &:= \{(l, d) \in L_{j,k} \ : \ l \cap l_{j,k} = \emptyset\}\\ L_{j,k,1/2} &:= \{(l, d) \in L_{j,k} \ : \ l \cap \{(m, n), (m', n')\} \neq \emptyset\}\\ L_{j,k,1} &:= \{(l, d) \in L_{j,k} \ : \ l \cap l_{j,k} \setminus \{(m, n), (m', n')\} \neq \emptyset\}\\ R_{j,k,0} &:= \{(l, d) \in R_{j,k} \ : \ l \cap l_{j,k} = \emptyset\}\\ R_{j,k,1/2} &:= \{(l, d) \in R_{j,k} \ : \ l \cap \{(m, n), (m', n')\} \neq \emptyset\}\\ R_{j,k,1} &:= \{(l, d) \in R_{j,k} \ : \ l \cap l_{j,k} \setminus \{(m, n), (m', n')\} \neq \emptyset\}. \end{aligned} \tag{106}$$

The dependency equation then becomes:

$$v_k = v_j + r_{j,k} \text{ with } r_{j,k} = |R_{j,k,1}| + \frac{1}{2}\left|R_{j,k,1/2}\right| - |L_{j,k,1}| - \frac{1}{2}\left|L_{j,k,1/2}\right|, \tag{107}$$

where j and k refer to adjacent cells such that $j, k \in \left\{0, 1, ..., N^2 - 1\right\}$. There will be

$$\sum_{j=0,...,N^2-1} |B_j| = 4 * (N-2)^2 + 3 * 4 * (N-2) + 2 * 4 = 4N(N-1) \tag{108}$$

unidirectional pairs and, discounting duplicate mirror pairs, we will end up with $2N(N-1)$ bidirectional pairs. These individual dependencies are defined in Eq. 107, while there are only N^2 free parameters $(v_j)_{j=0}^{N^2-1}$. This results in an overconditioned system, which is not solvable for an arbitrary set of contour line segments C.

2.12.2 Reconstructing a continuous function

So far we have constructed the a contour set C from a continuous function f, and represented these contour segments as dependencies between adjacent grid points. We will show that this representation produces function f', which deviates from f no more than unity:

$$|f(x, y) - f'(x, y)| \leq 1 \qquad (x, y) \in X. \tag{109}$$

We can then conclude, that by solving the Poisson's equation Eq. 91, we can convert sets of contours into scalar intensity maps in the ideal case where all contours

are complete. Such contours only end at the map boundaries, and otherwise form closed loops.

There is a solution $\boldsymbol{v} = (v_j)_{j=0}^{N^2-1}$ restricted by all dependencies defined in Eq. 107, supposing the contour line segments in C originate from a continuous function. An arbitrary continuous function can then be reconstructed from its contours, up to a precision set by the contour level spacing.

We only need to be interested in such line segments in C that actually intersect with the rectangular grid $\bigcup l_{j,k}$, since for a particular adjacent pair (j,k), all other line segments are contained in $F_{j,k}$, $R_{j,k,0}$, and $L_{j,k,0}$. Suppose we have a continuous function $f \ : \ X \mapsto \mathbb{R}$ where $X = [0, N-1]^2$. There is a set $S \subset X$ such that:

$$S = \{(x,y) \in X \ : \ f(x,y) \in \mathbb{Z}\}\,, \tag{110}$$

which constitute the contour lines of f. Over the contours is placed a grid, which is defined as:

$$G = \{(x,y) \in X \ : \ x \in \mathbb{Z} \text{ or } y \in \mathbb{Z}\}\,. \tag{111}$$

The intersection $S \cap G$ is closed, since both G and S are closed. The fact that S is closed follows from the continuity of f by contradiction:

Proof. Suppose that S is not closed. Then $X \setminus S$ is not open. Then for some $z \in X$ we have $f(z) \notin \mathbb{Z}$, and for every $\epsilon > 0$, there is another $z' \in X \setminus S$, $z' \neq z$, such that $f(z') \in \mathbb{Z}$ and $|z - z'| < \epsilon$. The lower limit for $|f(z') - f(z)| = e$ is a finite number $e \in (0,1)$, and thus f is not continuous. □

Typically $S \cap G$ will be a finite set of points, but there are also special cases with continuous regions in $S \cap G$, since f may have regions of constant value. We will represent $S \cap G$ as a finite union of closed non-overlapping sets:

$$S \cap G = \bigcup G_i \qquad G_i \text{ closed}, \qquad G_i \cap G_j = \emptyset \qquad i \neq j. \tag{112}$$

Since $S \cap G$ is closed and the partition is finite, the sets G_i are strictly separated by some finite distance $\epsilon > 0$, meaning that if

$$G_{i,\epsilon} = \{z \in X \ : \ \exists y \in G_i \text{ such that } |z - y| \leq \epsilon\}\,, \tag{113}$$

then for $j \neq i$, also $G_{i,\epsilon} \cap G_{j,\epsilon} = \emptyset$. In practice, the required ϵ may be very small, so that we may instead pre-define some fixed value for ϵ and measure distances between $\{G_i\}$ as:

$$d\,(G_i, G_j) = \min\{|g_i - g_j| \ | \ g_i \in G_i,\ g_j \in G_j\}\,. \tag{114}$$

We will then merge nearby subsets and form a new partition $\{G'_i\}_{i=1}^{i_{\max}}$ and link them to sets of the old partition using finite sets of integers, $\{C_i\}_{i=1}^{i_{\max}}$:

$$G'_i = \bigcup_{j \in C_i} G_j \qquad \forall j \ : \ \exists k \in C_i \text{ such that } (G_j, G_k) \leq \epsilon \tag{115}$$

$$d(G'_i, G'_l) > \epsilon \text{ for } i \neq l. \tag{116}$$

The required ϵ must be small enough so that $f(z)$ does no vary too much:

$$|z - y| \leq \epsilon \quad \Rightarrow \quad |f(z) - f(y)| < 1 \qquad \text{for } y, z \in X. \tag{117}$$

This ensures that we do not merge contours with different level, but do not allow two reference sets, G'_i and G'_j, to appear too close to each other. We may then construct hulls for each G'_i as:

$$H_i = \{z \in X \ : \ \exists y \in G'_i \text{ such that } |z - y| = \epsilon\}. \tag{118}$$

Each hull will encompass its reference set. When z goes around the hull H_i while we measure $f(z)$, we will experience an even number of transitions across an integer level $f_i \in \mathbb{Z}$:

$$z' \in G'_i \quad \Rightarrow \quad f(z') = f_i \in \mathbb{Z}. \tag{119}$$

The special case $H_i \cap \partial X \neq \emptyset$ has to be treated accordingly at the map boundaries. Otherwise the transitions across an integer boundary will split H_i into sets

$$(H_{i,1}, H_{i,2}, ..., H_{i,2n_i}). \tag{120}$$

The value of $f(z)$ alternates as:

$$z \in H_{i,2m} \quad \Rightarrow \quad f(z) \geq f_i \qquad z \in H_{i,2m+1} \quad \Rightarrow \quad f(z) \leq f_i, \tag{121}$$

and the adjacent sets will intersect in a single point:

$$H_{i,m} \cap H_{i,m+1} = \{h_m\}. \tag{122}$$

For cyclicity, we also define that $H_{i,2n_i+1} = H_{i,1}$. For non-adjacent m and p, the sets do not intersect:

$$H_{i,m} \cap H_{i,p} = \emptyset. \tag{123}$$

We require that the sets in 120 are closed and that they constitute the hull $\cup H_{i,p} = H_i$. We will also require that for $m = 0, 1, ..., 2n_i$:

$$\exists z_{2m} \in H_{i,2m} \text{ such that } f(z_{2m}) > f_i, \tag{124}$$

and

$$\exists z_{2m+1} \in H_{i,2m+1} \text{ such that } f(z_{2m+1}) < f_i. \tag{125}$$

In general case there there are only 2 such sets and there is only one possible pair $(H_{i,1}, H_{i,2})$ which satisfies the above requirements, and thus $n_i = 2$. In special

cases it is possible that $n_i = 0$, since the contour path or point may constitute a local extremum. If the contour appears at at saddle point of $f(z)$, then the contours will extend in four directions with $n_i = 4$. When $f(z)$ is not analytical, there may also be multiple choices for $(H_{i,m})_{m=1}^{n_i}$, since the function may have plateaus.

We are interested in the directions where $f(z) = f_i$, away from G_i'. In order to handle all special cases, we can pick an arbitrary point from each G_i', namely $g_i \in G_i'$, and it follows that $f(g_i) = f_i$. Then we can place n_i contour segments which begin at g_i and are directed to each hull divider point $h_{i,m}$, $m = 1, ..., n_i$ as:

$$d_{i,m} := \sigma_{i,m} \frac{h_{i,m} - g_i}{|h_{i,m} - g_i|}, \qquad l_{i,m} := \{g_i + \alpha \epsilon d_{i,m} \ : \ \alpha \in [0,1]\}, \text{ and} \tag{126}$$

$$\sigma_{i,m} \in \{-1, +1\}. \tag{127}$$

When the signs $\{\sigma_{i,m}\}$ are set appropriately, the contour set

$$C := \{(l_{i,m}, d_{i,m}) \ : \ i = 1, ..., |\{G_i'\}|, m = 1, ..., n_i\} \tag{128}$$

will produce the required dependencies for Eq. 107 for reconstructing f. We assume that the boundaries are not grounded, so that the potential depends only on neighbouring cells within the map, and the average potential of the map is a free parameter.

2.12.3 Poisson's equation from dependencies

So far we can an overconditioned system with more dependencies than points. When the contours are complete, these dependencies are redundant. When we remove the redundancy, we end up with a linear system with only one degree of freedom. This degree of freedom allows us to choose the reference potential. We will set it so that the potential is zero outside the solar disk.

We can shrink the number of degrees of freedom by mutually summing those instances of Eq. 107 which share the left side and the index k. This results in a system of N^2 equations, which is equivalent to the Poisson's equation with 5-point stencil in Eq. 92. When we are outside any boundary, so that $x, y \in \{2, 3, ..., N-1\}$, the stencil is written as:

$$\nabla^2_{x,y} = \frac{a_{x-1,y} + a_{x+1,y} + a_{x,y-1} + a_{x,y+1} - 4a_{x,y}}{4}. \tag{129}$$

Around boundary points are less than four neighbours, so that

$$\nabla^2_{x,y} = \frac{a_{x-1,y} + a_{x+1,y} + a_{x,y-1} + a_{x,y+1} - 4a_{x,y}}{4}. \tag{130}$$

When at a boundary, we have two options. If the boundary is grounded, we will

treat the nodes outside the box as zeros. For example:

$$\nabla^2_{1,y} = \frac{a_{2,y} + a_{x,y-1} + a_{x,y+1} - 4a_{1,y}}{4} \quad \text{(grounded boundary)}. \tag{131}$$

This will force the potential to zero at the map boundary, and this will be done when the boundary is at the background where the sky is dark. There are also ungrounded, floating boundaries. Then we ignore the nodes outside the box and set:

$$\nabla^2_{1,y} = \frac{a_{2,y} + a_{x,y-1} + a_{x,y+1} - 3a_{1,y}}{3} \quad \text{(floating boundary)}. \tag{132}$$

Floating boundary is used initially when we want to give the contours as much freedom as possible. Once the background is fixed, we will still have floating boundaries where the solar disk extend beyond the map and is cropped.

When the contours are complete, the Poisson's equation results in a potential with transitions of ± 1 whenever there is a contour. The potential is locally constant everywhere else (Fig. 30a). The actual contours obtained from image recognition are incomplete with missing segments. Given a linear system with incomplete contours C_b and equivalent charge distribution $\boldsymbol{y}_\mathrm{b}$, the resulting scalar field will be $\boldsymbol{x}_\mathrm{b}$ as:

$$A\boldsymbol{x}_\mathrm{b} = \boldsymbol{y}_\mathrm{b} \quad \Rightarrow \quad \boldsymbol{x}_\mathrm{b} = A^{-1}\boldsymbol{y}_\mathrm{b}. \tag{133}$$

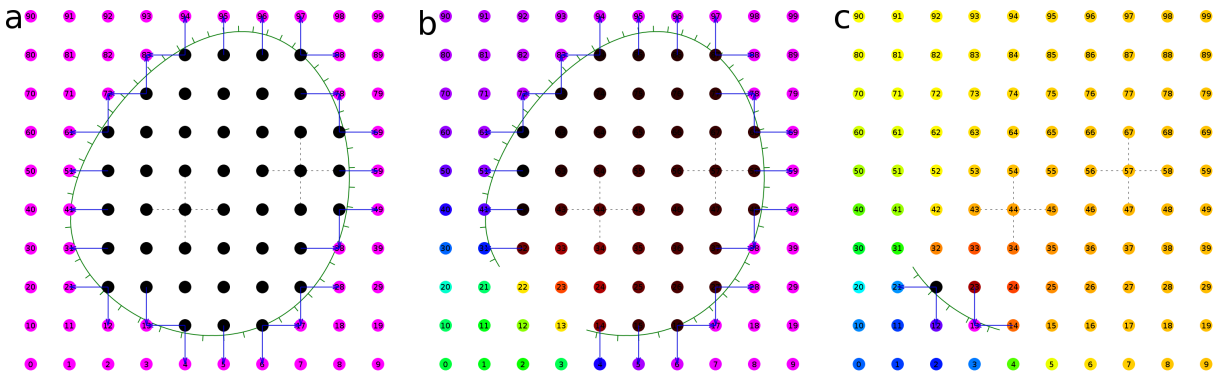

Figure 30: *Grid points are marked with spheres, whose colours indicate their scalar potential values. The colours black, red, yellow, green, blue, and violet refer to subsequently increasing potentials. The scalar field arises from dipoles, which are denoted as arrows connecting adjacent spheres. These dipoles appear when a contour line passes through pairs of adjacent points. a) With a complete contour, the resulting set of dipoles results in a flat potential field with sharp transition when crossing the contour. b) When the contour is broken, there is a smooth transition in the potential. c) The missing part of the broken contour as a perturbation to the complete contour. The system in picture b is a superposition of systems a and $-c$, namely, $b = a - c$. At distance to the source dipoles, the perturbation is very weak and smooth. Our goal is to identify and eliminate this perturbation by considering adjacent potential values at the distance and forcing them equal.*

The charge distribution is constructed as a sum which arises from intersections with contours and adjacent pairs:

$$\boldsymbol{y}_\mathrm{b} = \sum_{\left(\begin{smallmatrix}(i,j)\text{ adjacent}\\ c \in C\end{smallmatrix}\right)} \left(|L_{i,j,c}| - |R_{i,j,c}|\right)\left(\hat{\boldsymbol{e}}_i - \hat{\boldsymbol{e}}_j\right). \tag{134}$$

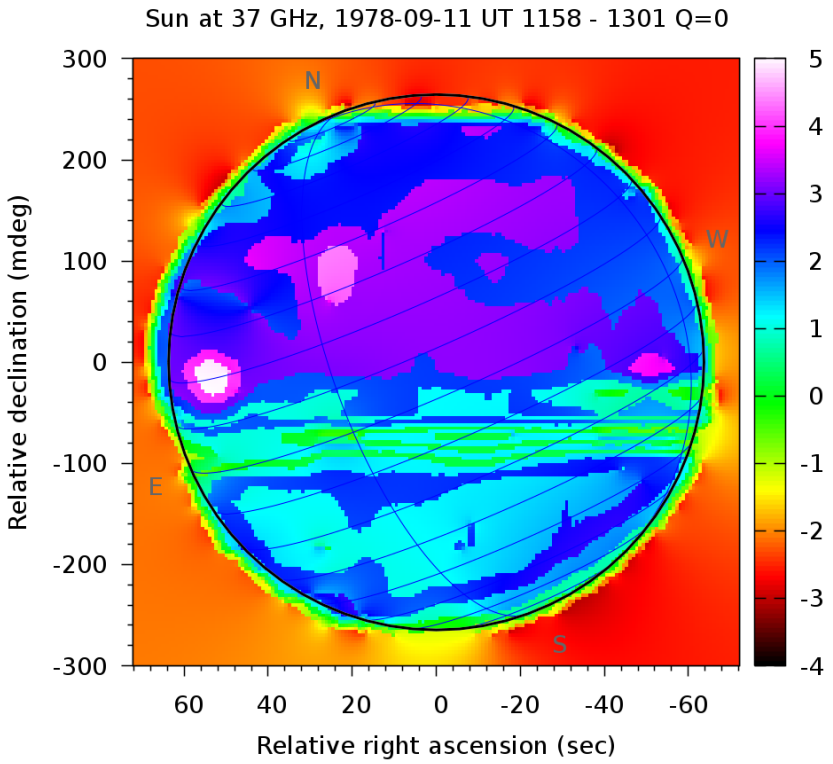

Figure 31: Simple Poisson's solution for an incomplete set of contours.

We need a generalised unit vector $\boldsymbol{e}_j$ to have its j:th unity and the rest zero. The sets $L_{i,j,c}$, $R_{i,j,c}$, and $M_{i,j,c}$ are a partition of G_i - $G_j \cap c$, meaning that:

$$L_{i,j,c} \cup R_{i,j,c} \cup M_{i,j,c} = G_i \text{ - } G_j \cap c, \tag{135}$$

and $L_{i,j,c} \cap R_{i,j,c} = L_{i,j,c} \cap M_{i,j,c} = R_{i,j,c} \cap M_{i,j,c} = \emptyset$. The sets $L_{i,j,c}$, $R_{i,j,c}$, and $M_{i,j,c}$ contain the points in which a contour c, equipped with a sign or direction in order to mark the direction of gradient in the intensity field, does intersects with a line segment G_i - G_j. Important values are $|L_{i,j,c}|$ and $|R_{i,j,c}|$, which represent the number of points in the corresponding sets. Each intersection point is counted into exactly one of these sets $L_{i,j,c}$, $R_{i,j,c}$, and $M_{i,j,c}$, depending on whether the contour line travels from left to right, right to left, or straight, with respect to the line segment G_i - G_j. There are typically zero and sometimes one such point in a particular set, so that $|L_{i,j,c}| - |R_{i,j,c}| \in \{-1, 0, 1\}$, but there may be several if the contour is making a sharp turn. A simple solution to the Poisson's equation will produce the correct intensity field, supposing the contours are complete. If the contours have missing or misinterpreted segments, they will appear as gradients in the scalar field, as displayed in Fig. 31.

The charge can also be expressed, referring to Eq. 107, as

$$y_i = \sum_{j \text{ adjacent to } i} r_{i,j}. \tag{136}$$

2.12.4 Eliminating the perturbation

We have now solved the Poisson's equation $A\boldsymbol{x}_\mathrm{b} = \boldsymbol{y}_\mathrm{b}$ for the known contour charge distribution $\boldsymbol{y}_\mathrm{b}$ and obtained the scalar potential field $\boldsymbol{x}_\mathrm{b}$, which represents the observed radio intensity on a solar map. The matrix A is the Laplace operator in a discrete rectangular grid. If all the contours were complete, $\boldsymbol{y}_\mathrm{b}$ would contain a consistent intensity field, as shown earlier in 2.12. The intensity will have discontiunities of unit step situated at contours lines. Elsewhere the intensity has locally constant integer value. However, typical digitised maps will contain broken contours, which will result in gradients in $\boldsymbol{x}_\mathrm{b}$.

The contour charge distribution $\boldsymbol{y}_\mathrm{b}$ is a sum of dipoles over adjacent pairs of grid points, constructed from the contour set $\mathcal{C}_\mathrm{b}$ as:

$$\boldsymbol{y}_\mathrm{b} = q\left(\mathcal{C}_\mathrm{b}\right) \quad \text{(see also Eq. 134 and 136).} \tag{137}$$

We can think that $A\boldsymbol{x}_\mathrm{b} = \boldsymbol{y}_\mathrm{b}$ is actually a superposition of two Poisson's equations:

$$\boldsymbol{y}_\mathrm{b} = \boldsymbol{y}_\mathrm{a} - \boldsymbol{y}_\mathrm{c} \qquad \boldsymbol{x}_\mathrm{b} = \boldsymbol{x}_\mathrm{a} - \boldsymbol{x}_\mathrm{c}, \tag{138}$$

so that

$$A\left(\boldsymbol{x}_\mathrm{a} - \boldsymbol{x}_\mathrm{c}\right) = \boldsymbol{y}_\mathrm{a} - \boldsymbol{y}_\mathrm{c} \text{ with } \boldsymbol{y}_\mathrm{b} = \boldsymbol{y}_\mathrm{a} - \boldsymbol{y}_\mathrm{c} \text{ and } \boldsymbol{x}_\mathrm{b} = \boldsymbol{x}_\mathrm{a} - \boldsymbol{x}_\mathrm{c}. \tag{139}$$

In order to define such a superposition, we will require that $\boldsymbol{y}_\mathrm{a}$ originates from a complete contour set $\mathcal{C}_\mathrm{a}$, such that $\boldsymbol{y}_\mathrm{a} = q\left(\mathcal{C}_\mathrm{a}\right)$. The complete contour set can chosen arbitrarily, as long as $\mathcal{C}_\mathrm{a} \supseteq \mathcal{C}_\mathrm{b}$. We can imagine that all the broken contours are joined in a neat way, but theoretically this is not required here.

The other component in the superposition, $\boldsymbol{y}_\mathrm{c}$, likewise originates from a contour set $\mathcal{C}_\mathrm{c}$, as $\boldsymbol{y}_\mathrm{c} = q\left(\mathcal{C}_\mathrm{c}\right)$. We set $\mathcal{C}_\mathrm{c} := \mathrm{closure}\left(\mathcal{C}_\mathrm{a} \setminus \mathcal{C}_\mathrm{b}\right)$, and it follows that $\mathcal{C}_\mathrm{a} = \mathcal{C}_\mathrm{b} \cup \mathcal{C}_\mathrm{c}$ and $\mathcal{L}_1\left(\mathcal{C}_\mathrm{b} \cap \mathcal{C}_\mathrm{c}\right) = 0$. The latter requirement means that the contour sets intersect only in discrete points, so that the line measure $\mathcal{L}_1$, Lebesgue measure of dimension one [54], is zero for the intersection. Both sets $\mathcal{C}_\mathrm{b}$ and $\mathcal{C}_\mathrm{c}$ have to be closed, so that $\mathcal{C}_\mathrm{b} \cap \mathcal{C}_\mathrm{c} \neq \emptyset$ unless $\mathcal{C}_\mathrm{b}$ already contains a complete set of contours. Eq. 139 then follows from the definition of q (Eq. 137).

We can think that $\boldsymbol{y}_\mathrm{c}$ is a perturbation to the complete contour problem $A\boldsymbol{x}_\mathrm{a} = \boldsymbol{y}_\mathrm{a}$, which has no gradients in $\boldsymbol{x}_\mathrm{a}$. We do not know the exact value of the perturbation, since it is arbitrary, but we can start by masking those areas where the perturbation is not significant. We will assign a number associated with every pair of adjacent grid points:

$$\forall a, b \in G \text{ such that } d(a,b) = 1 \;:\; e(a,b) = \left|G(b) - G(a) - C(b,a)\right|. \tag{140}$$

Here $C(b,a) = r_{b,a}$ counts how many contours flow between the pair of adjacent grid points, as in Eq. 107.

We will get a list of the adjacent pairs $((a_i, b_i))_{i=1}^{2N(N-1)}$, sorted in ascending order for $e(a_i, b_i)$. Due to symmetry, we can require that $a_i < b_i$. We will begin with the first item in the list, that is, (a_1, b_1), and substitute

$$x_{b_1} := x_{a_1} + C(b_1, a_1) \text{ into the linear system } A\boldsymbol{x}_\mathrm{b} = \boldsymbol{y}_\mathrm{b}. \tag{141}$$

This will lead to a new linear system $A^{(1)}\boldsymbol{x}^{(1)} = \boldsymbol{y}^{(1)}$, which has one less degrees of freedom. This new linear system is mapped to the original as:

$$x_i^{(1)} = \begin{cases} x_i & \text{when } i < b_1 \\ x_{i-1} & \text{otherwise} \end{cases}. \tag{142}$$

In the iteration, $A^{(1)}$ and $\boldsymbol{y}^{(1)}$ are obtained from A and $\boldsymbol{y}_\mathrm{b}$ with the transforms associated in Eq. 141. By definition, $C(b_1, a_1) = y_{b_1} - y_{a_1}$. The linear system becomes:

$$y_k = \sum_{j=1}^{N} A_{k,j}x_j = \sum_{j=1,\, j\neq b_1}^{N} A_{k,j}x_j + A_{k,b_1}\left(x_{a_1} + y_{b_1} - y_{a_1}\right). \tag{143}$$

This can be represented as an overconditioned $N \times (N-1)$ system for $k = 1, 2, ..., N$,

$$y_k - A_{k,b_1}\left(y_{b_1} - y_{a_1}\right) = \sum_{j=1}^{N-1} A'_{k,j}x_j^{(1)}, \tag{144}$$

where

$$A'_{k,j} = \begin{cases} A_{k,j} & \text{when } j < b_1 \text{ and} j \neq a_1 \\ A_{k,a_1} + A_{k,b_1} & \text{when } j = a_1 \\ A_{k,j+1} & \text{when } j > b_1 \end{cases}. \tag{145}$$

In order to get a solvable linear system, we have to reduce the amount of degrees of freedom in Eq. 144. We will average the rows $(k = a_1)$ and $(k = b_1)$. This will define the a_1:th row in the new linear system $A^{(1)}\boldsymbol{x}^{(1)} = \boldsymbol{y}^{(1)}$ as:

$$y_{a_1}^{(1)} = \frac{1}{2}\left[y_{a_1} + y_{b_1} - \left(A_{a_1,b_1} + A_{b_1,b_1}\right)\left(y_{b_1} - y_{a_1}\right)\right]. \tag{146}$$

$$A_{i,j}^{(1)} = \begin{cases} A'_{i,j} & \text{when } i < b_1 \text{ and } i \neq a_1 \\ \frac{1}{2}\left(A'_{a_1,j} + A'_{b_1,j}\right) & \text{when } i = a_1 \\ A'_{i+1,j} & \text{otherwise} \end{cases}. \tag{147}$$

We can process the first m pairs in the list $((a,b)_i)_{i=1}^{2N(N-1)}$ and perform the substitutions:

$$x_{b_1}^{(j)} := x_{a_1}^{(j-1)} + C(b_j, a_j) \tag{148}$$

into the linear system:

$$A^{(j-1)}\boldsymbol{x}^{(j-1)} = \boldsymbol{y}^{(j-1)} \quad \text{for} \quad j = 2, 3, ..., m, \tag{149}$$

in a similar manner as in Eq. 141-147. Finally we will solve the linear system and assign new intensity values for the grid points.

Areas of low perturbation will form integral crystals in the scalar field. Within a particular crystal at the m:th iteration loop, denoted with the c:th row in the linear system, the potential is described with a single variable $x_c^{(m)}$. For each grid point x_j within the crystal, the potential has a fixed offset to the crystal potential as $x_j = x_c^{(m)} + o_j$. In the subsequent steps of solving the shrinking linear system, we will have larger and fewer crystals left. Ideally a single crystal would grow to cover almost the whole map, and leave only very small crystals in limited regions. These limited regions will then contain the missing contour segments $\mathcal{C}_\mathrm{c}$. An example of this process is given in Fig. 32, where the regions with unified color represent areas which are described using a single variable. The images from top to bottom represent the dimensionality of the linear system after subsequent step of m substitutions and then solving the linear system $A^{(m)}\boldsymbol{x}^{(m)} = \boldsymbol{y}^{(m)}$.

At some point, too drastic changes to the potential field would arise by further reducing degrees of freedom, so we stop the iteration once all the remaining adjacent pairs are exhibiting too much perturbation. Finally we will round all the grid values into nearest integers in order to eliminate any gradients from the field.

As a result, we will obtain a clean map, since almost all the perturbations from missing contour segments are eliminated (Fig. 33). Sometimes there are multiple integer values outside the visual solar disk where it should be black sky. We will select the lowest integer, still containing reasonably many grid points, to represent the black sky, zero contour level, and cut off any remaining negative artifacts. The parameters for fine-tuning this process are listed in Appendix. B.24. We can change how many substitution steps m to take by setting the parameter *shrink_factor*, *shrink_dim*, and *contour_compress_tol*.

2.13 Bright and dim regions

Having obtained the scalar field $v(x, y)$, we are interested in the bright and dim regions. This was not an original requiremement of this work, but still very useful later when we need quality assessment. The first step is to calculate statistical

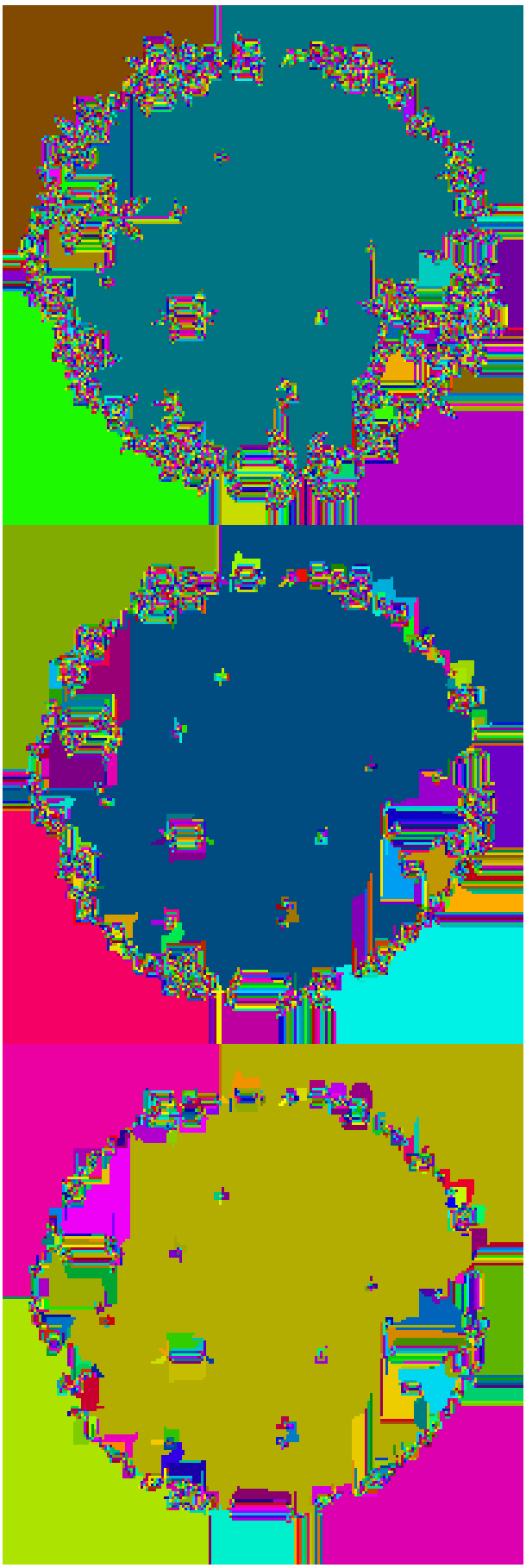

Figure 32: Iterative dimensionality reduction. Images from top to bottom represent subsequent steps. Adjacent cells with low level of perturbation and good agreement are joined by performing appropriate substitution to the Poisson's equation. Perturbation remains at the perimeter of the solar disk.

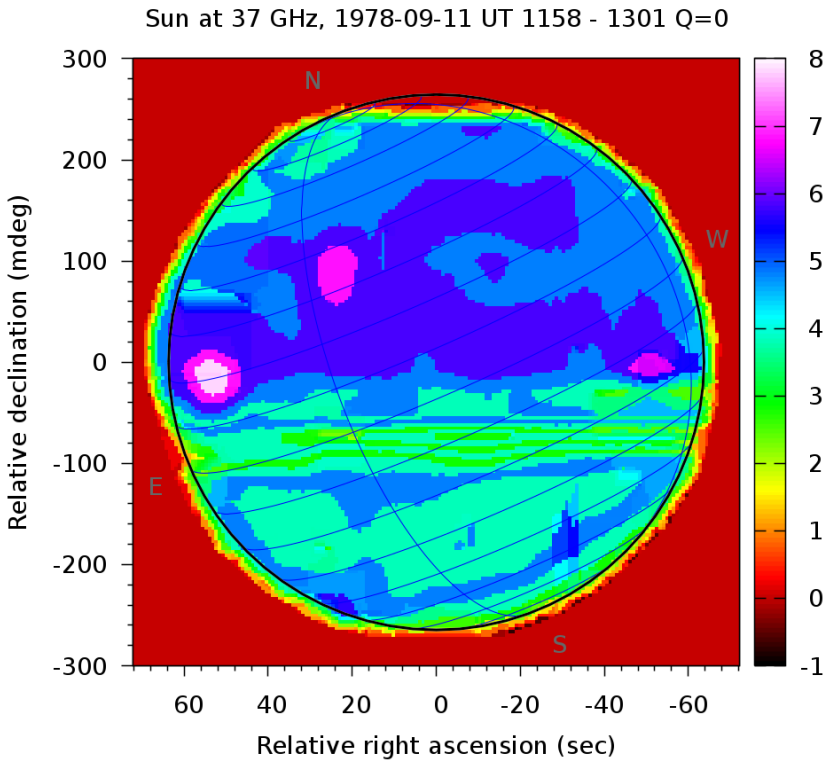

Figure 33: *Intensity map after dimensionality reduction and proper selection of the background level.*

average and deviation of the scalar values at the grid points $\{v_{i,j}\}$:

$$\langle v\rangle = \frac{1}{N^2}\sum_{i,j=0,\dots,N-1} v_{i,j} \qquad \langle v^2\rangle = \frac{1}{N^2}\sum_{i,j=0,\dots,N-1} v_{i,j}^2. \tag{150}$$

The standard deviation is then calculates as:

$$\sigma = \sqrt{\langle v^2\rangle - \langle v\rangle^2}. \tag{151}$$

The regions where intensity $v \in [\langle v\rangle - 2\sigma, \langle v\rangle + 2\sigma]$ are considered as quiet Sun surface. We will iterate the process by initializing first:

$$\langle v\rangle_0 := \langle v\rangle \quad \text{and} \quad \langle v^2\rangle_0 := \langle v^2\rangle \qquad \sigma_0 := \sigma. \tag{152}$$

Then we will iterate starting from $k = 0$ and define a new quiet Sun surface:

$$Q_{k+1} := \{(i,j) \ : \ v_{i,j} \in [\langle v\rangle_k - 2\sigma_k, \langle v\rangle_k + 2\sigma_k]\}. \tag{153}$$

We calculate new averages from quiet Sun surface only:

$$\langle v\rangle_{k+1} = \frac{1}{|Q_{k+1}|}\sum_{(i,j)\in Q_{k+1}} v_{i,j} \qquad \langle v^2\rangle_{k+1} = \frac{1}{|Q_{k+1}|}\sum_{(i,j)\in Q_{k+1}} v_{i,j}^2. \tag{154}$$

A new value for standard deviation comes from these iterated averages:

$$\sigma_{k+1} = \sqrt{\langle v^2\rangle_{k+1} - \langle v\rangle_{k+1}^2}. \tag{155}$$

For reasonable data, the iteration will converge quickly, typically with $k = 6$, so that we can set the QSL at $\langle v \rangle_6$. Everything below $v_{\min} := \langle v \rangle_6 - 2\sigma_6$ will be treated as a dim region, and everything above $v_{\max} := \langle v \rangle_6 + 2\sigma_6$ will be treated as bright region.

There is an option in the code to force the QSL halfway between two adjacent contours. This requires that we proceed the iteration by using a substitute value for $\langle v \rangle_k$:

$$\mu_k := \lfloor \langle v \rangle_k \rfloor + \frac{1}{2}. \tag{156}$$

We can also give μ_k as an external parameter to the code. In the following iterations, we calculate the standard deviation with respect to μ_k instead:

$$|Q_k| \, \sigma_k^2 = \sum_{(i,j) \in Q_k} (v_{i,j} - \mu_k)^2 \tag{157}$$

$$= \sum_{(i,j) \in Q_k} \left(v_{i,j}^2 - 2\mu_k v_{i,j} + \mu_k^2 \right). \tag{158}$$

We can still use the precalculated $\langle v^2 \rangle_k$ and $\langle v \rangle_k$ so that only one loop is required.

$$\sigma_k = \sqrt{\langle v_{i,j}^2 \rangle + \mu_k \left(\mu_k - 2\langle v \rangle_k \right)}. \tag{159}$$

Another option is to give $v_{\min}$and $v_{\max}$ as external parameters.

2.14 Recursive bright and dim regions

A bright or dim regions may constitute a single extremum or it may have fine structure. For this purpose, a recursive tree structure is useful. When an bright region has several brightness maxima, these individual extrema will be included as subregions in the said larger region. We are only interested in the roots and leaves of such a hierarchy. Leaves are such regions which have no subregions, while roots are not subregions of any other region. The intermediate regions are not shown on maps nor listed in the JSON output, but they limit the size of their subregions. Any region always covers its subregions, and higher level region is formed upon contact.

We aim to build unified regions from the grid points with $v_{i,j} < v_{\min}$ and $v_{i,j} > v_{\max}$. Assuming the intensity is continuous, grid points adjacent to an intensity maximum are expected to be more intense than the surroundings. Since the grid intensity values $\{v_{i,j}\}$arise from contours, adjacent grid points typically have equal values unless there is a contour in between. Even when between to contour curves, it would be better to have slightly larger intensities where the nearby maximum is closer. In order to have such gradients in the intensity map, we will first apply a low pass filter to the data. The low pass filter will transform a sinusoidal pattern of wavenumber k and unity amplitude into a pattern of amplitude $e^{-\ln(2)k/k_{1/2}}$and

equal phase. Since the grid spacing is generally non-isometric, care has to be taken in order to obtain isometric filtering.

Having filtered the scalar field, we will denote the filtered intensity values with $v'_{i,j}$. We will build a sorted list $(S_k)_{k=1}^{N^2}$ of grid points:

$$S_k = (v_k, i_k, j_k, \sigma_k) \text{ such that } v_{i_k,j_k} = v_k, \tag{160}$$

and σ_k here is the sign of the grid point. For bright regions it is $\sigma_k = +1$if $v_k > v_{\mathrm{QSL}}$, and for dim regions it is $\sigma_k = -1$. The order in (S_k) satisfies:

$$k \leq l \quad \Leftrightarrow \quad |v_k - v_{\mathrm{QSL}}| \leq |v_l - v_{\mathrm{QSL}}| \,. \tag{161}$$

It follows that as we pick elements from (S_k), starting from the first $k = 1$, we will first pick an extremum and second either another extremum or a grid point which is adjacent to the first extremum. We will keep track of how the forming regions are connected. If a picked S_k is not adjacent to any known extremum with the same sign, we will create a new extremum:

$$E_s := (S_k) \qquad \text{(leaf or root extremum region)}. \tag{162}$$

This new region may later become a leaf subregion if it is connected to another region. Otherwise it will be remain a root. If, instead, S_k is adjacent to one particular extremum E_t, we append S_k to E_t. In the case there are several adjacent extremum regions E_*, E_{**}, ..., with compatible sign, we will merge them into a new extremum.

$$E_s := (E_*, E_{**}, ..., S_k) \,\text{(branch or root extremum region)}. \tag{163}$$

The new region is initially a root, but may become a branch if more connections are found. We thus gradually build a recursive tree structure of bright and dim regions.

When all the non-quiet grid points are processed, we will finally calculate statistics of the obtained regions. That will include total power, total area, centre of mass, average intensity, and peak intensity. The same quantities can be calculated from both the low pass filtered data $v'_{i,j}$ and from the raw data $v_{i,j}$. The data is given in Java Script Object Notation (JSON) format. Power is the relative intensity integrated over the region area. It always calculated relative to QSL. Power is given as nano steradians QSL as observed from Earth. The final JSON output will only include roots and leaves so at this point the hierarchy must be flattened.

2.15 Measuring output quality

We cannot supervise the output, since any correct output should be manually verified. We are left with assessing the quality based on statistics. We can determine how long the identified paths generally are. Average path length is one criterion for output quality. There are more measures available from different

stages of the image processing. Good output should contain clear bright and dim regions with no folding defects and preferably no subregions. We can also study the convergence in the Poisson solver. IThe parameters used for calculating the quality are listed below, with coefficients representing their weight.

- Average path length as a number of reference points $*(+10)$.
- Final degrees of freedom after Poisson solver dimensionality reduction stops $*(-1)$.
- Amount of contours for which is not possible to determine the sign based on gradient indicators only $*(-20)$.
- Amount of gradient indicators detected $*(+1)$.
- Amount of gradient indicators which are in conflict with the majority for a particular contour $*(-3)$.
- Amount of root regions, either bright or dim $*(+1000)$.
- Amount of leaf regions, either bright or dim $*(-300)$.
- Total absolute power of the region, as nano steradians QSL seen from Earth $*(+5)$.
- Whether the compass rose or similar feature is detected (1) or not (0) $*(+5000)$.

The used weights for map quality are subject to change. For example, the folding defect detection could be improved. We could base the quality on how many contours have different sign compared to a corresponding envelope contour.

2.16 Parameter optimization for maximizing output quality

The code now uses $n \approx 100$ parameters, out of which many are sensitive to map quality. Some are related to debugging, development, documentation, and testing. There are also non-sensitive parameters which roughly restrict the layout. All parameters are changeable from command line. We use a set of parameters which reside in the parameter space:

$$(p_1, p_2, ..., p_n) = \boldsymbol{p} \in \mathbb{P}. \tag{164}$$

The parameter space $\mathbb{P}$ is supplied to the module which performs the *sunmap* digitisation algorithms. The parameter vector $\boldsymbol{p}$ contains boolean, integer and floating point elements. Suppose p_i contains integer or floating point values. Then there is a_i and b_i supplied in the parameter set definition, and we restrict that:

$$a_i \leq p_i \leq b_i. \tag{165}$$

Optionally, also c_i is given, which defines the spacing between possible values. For some parameters, it is possible to assign only even or only odd numbers. With the framework, any reasonable $c_i > 0$ is supported. The parameter space $[a_i, b_i]$

is divided to n_i intervals of equal length, based on the spacing c_i as:

$$n_i := \left\lfloor \frac{b_i - a_i}{c_i} + \frac{1}{2} \right\rfloor , \tag{166}$$

so that allowed parameter values have:

$$p_i = a_i + \frac{b_i - a_i}{n_i} * j \qquad \text{for some} \quad j \in \{0, 1, ..., n_i\} . \tag{167}$$

Having now defined the parameter space, the default action is to run the *sunmap* digitisation code with the pre-defined values for p_i. This configuration is originally found, with basic reasoning as well as trial and error, to work reasonably well with a particular map. Then sensible values for boundaries a_i and b_i were given, and some critical parameters were chosen for intensive optimization. For these particular $n = 31$ parameters, an evolutionary scheme was developed. It consists of a priority queue $\left(\boldsymbol{p}^{(j)}\right)_{j=1}^{M}$which is sorted according to the quality $Q\left(\boldsymbol{p}^{(j)}\right)$ of the target map obtained with configuration $\boldsymbol{p}^{(j)}$:

$$j \leq k \quad \Leftrightarrow \quad Q\left(\boldsymbol{p}^{(j)}\right) \geq Q\left(\boldsymbol{p}^{(k)}\right) . \tag{168}$$

The evolutionary optimization selects one of the following operations to be performed at each step:

1. Pick a new parameter configuration randomly as $\boldsymbol{p} \in \mathbb{P}$ (randomization).
2. Randomly pick an existing configuration $\boldsymbol{p}^{(j)}$ from the queue, and slightly modify one or more of the parameters $\left\{p_r^{(j)} \ : \ r \in \{1, 2, ..., n\}\right\}$ in it (mutation).
3. Randomly pick two existing configurations $\boldsymbol{p}^{(j)}$ and $\boldsymbol{p}^{(k)}$, $j \neq k$, and generate a new configuration $\boldsymbol{p}^{(m)}$ by recombination.
4. Perform recombination followed by mutation.

In recombination we randomly assign:

$$p_r^{(m)} \in \left\{p_r^{(j)}, p_r^{(k)}\right\} \tag{169}$$

for each $r = 1, 2, ..., n$. The map is processed again with $\boldsymbol{p}_m$, and the new configuration added into to priority queue according to $Q\left(\boldsymbol{p}_m\right)$. We first run randomization until the queue is suitably long, e.g. $M = 40$, and afterwards we choose randombly whether to mutate, recombine, or do both. Altering the propabilities different operators will definitely change the rate at which map quality improves for $\boldsymbol{p}_1$. However, fine-tuning the optimization process was not part of this study. Quantitative analysis of the quality convergence will be part of future work. In

practice acceptable map quality was achieved after a few hundred optimization steps, and that took several hours.

Single map digitisation typically requires 40 seconds of CPU time on typical workstations, as of spring 2018, in Aalto University. The workstations had processors *Xeon E3-1230v5 3.40GHz 8MB Turbo Boost* with eight CPUs and Linux for x86-64 architecture. Parameter optimization was run for 14 target maps, which were chosen arbitrarily with some emphasis on trying to treat different contour distributions. The optimization run uses one thread and process per map. Having obtained 14 parameter configurations which handle a target map reasonably, all maps 1028 of the data set were processed with each of these 14 parameters. In order to complete the run in reasonable time, a few hours, the load was distributed to several processes which all run one thread.

The emphasis was to utilize only these computational resources which were idle. Workload distribution was done with *Bash* scripts. They first check that whether soneone is logged in on the workstations, and immediately quit unless no one was logged in. Before starting to process a map, the code always waits for two seconds and measures the computational load on the workstation. If any load above the level expected from idle system processes was detected, the processing was delayed. The same is done if someone is logged in. The *sunmap* process internally calls finger and *mpstat* programs present in the Linux distribution, and processes their output in order to extract the information about whether the workstation is idle or not. No information was stored or kept in memory once the child processes were waited. If the workstation was idle, the *sunmap* code waited for some minutes before re-checking the idleness. When processing foreground, this option can be deactivated with argument *nice=0* to the *sunmap* program.

Each available workstation was utilized with four CPUs for the *sunmap* code. The results were stored to the common network filesystem, and contained following files for each map:

- Log output of the run.
- .metadata file required by present Metsähovi scripts.
- .json file which contained any metadata produced by the sunmap process.
- .map file which contains the scalar intensity field in a matrix form.
- rendered .png picture which shows the solar disk and available data.
- rendered .png picture which features a Carrington map.

Each map was given an unique ID, which by default is the unix time at the beginning of data. Care was taken not to allow to processes to create a map with same ID, so that the subsequent ID was given when to contour plots shared the beginning timestamp of data.

Out of the 14 different outputs of a particular contour maps, the one with best quality was delivered to Metsähovi.

2.17 Interpolating modern maps

The modern maps contain exact antenna direction information, which show irregularities (Fig. 34). They need to be resampled before they can be analysed for bright and dim regions. There are multiple options for resampling, but in Fig. 34 the map area has been sliced into 10×10 squares. Each square is a finite element, containing a polynomial function:

$$\sum_{a,b} C_{a,b} x^a y^b \qquad \text{where } a, b \in \{0, 1, ..., 7\} \text{ and } a + b \leq 7. \tag{170}$$

The boundaries of the squares are required to be continuous and twice differentiable. The resampling can be done with various number of rectangles or other polygons. Increasing the number of polygon segments will add more degrees of freedom into the interpolation, thus allowing the resampled function to match the original data. Increasing the contiunity requirement by having more continuous differentials however, will reduce degrees of freedom. If too many degrees of freedom are given, the interpolated map will overshoot between individual sweeps. If the degrees of freedom are too few, higher spatial frequencies will be cut off, and we end up with ringing artefacts. Fig. 34 represents good balance. The goal is to find such interpolation method that will minimise the amount of disturbances generated in sparse data sets. For example in Fig. 5c, the sweep direction is still visible, which suggests that the interpolation used is still not perfect. In future work, this can be tested by generating a suitable continuous function and resampling it with an irregular sweep pattern. After resampling, we will interpolate these samples and obtain a new continuous function. This should well resemble the original function. If we translate or rotate the sampling pattern, the interpolated result should not change.

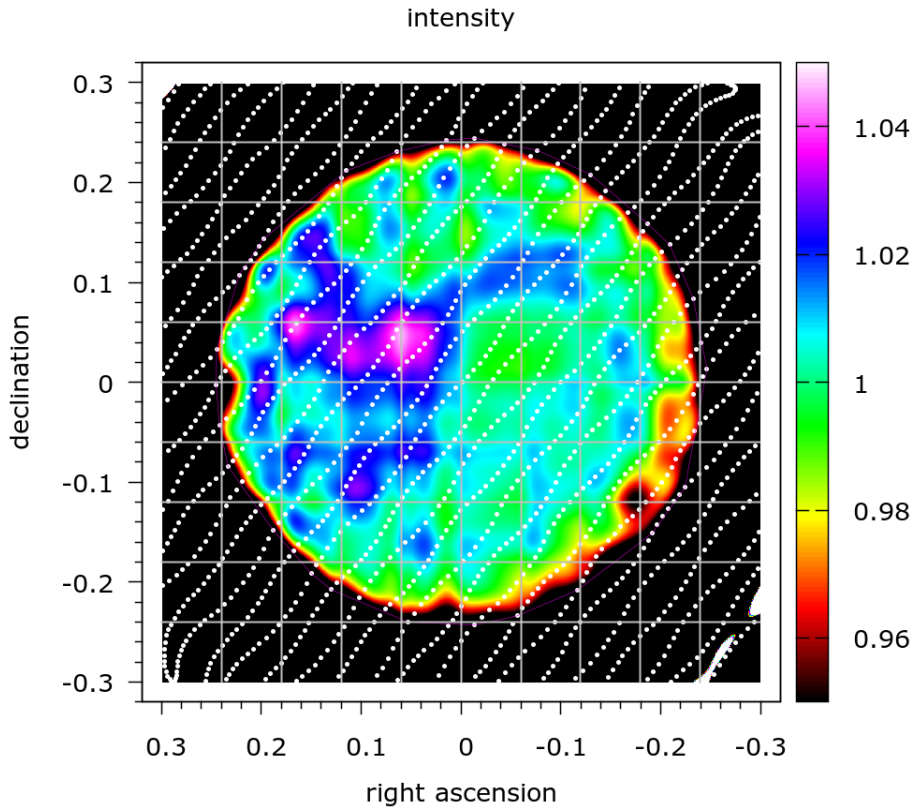

Figure 34: *Example of a modern map, with antenna samples and an interpolated heatmap. The gray grid defines the finite elements.*

3 Results

Out of 1028 maps, 965 were digitized with the requirement that layout was detected properly. In addition to rendered contour graphics these 1028 maps included 19 matrix printer maps from year 1987. There is not yet a parameter set available for processing this graphics. Such a set would be necessary for processing the following year, 1988, for which the mechanically plotted contour graphics was no longer used. For year 1987, the maps also exist in mechanically plotted format. There are also seven maps from the past decade, year 1979, which where experimentally printed with a matrix printer once the technology was adopted. Of the remaining 37 failures, three were due to too strict tolerance in the box aspect ratio. All the rest were failed due to unsuitable cropping or broken box lines. All of these failures will be fixed as the continuous fine-tuning of the software proceeds further. A significant part of the development process has been the extension of the parameter set to treat various special cases and standards.

When individual observations are considered, there are 536 maps in the contour format, out of which 453 were full solar disks. For the remaining 83 maps, either the intention was to observe a close-up of an interesting region, or the associated full solar map is lost. The criterion whether a map is a full solar map or a close-up is based on whether at least 200° of the solar terminator is visible. The observational activity is distributed as in Fig. 35. There were also 223 scanned close-up maps which originated from a larger data set, so that not the whole data set was rendered as contours. 206 scanned maps were duplicates in the sense that same data set was mechanically plotted with different settings. It is sufficient to digitise one of these maps.

Two people have independently viewed the output maps in Metsähovi Radio Observatory. They have recognised 562 individual maps in total. They have omitted obvious duplicates and redundant maps, but have sometimes included multiple renderings of the same observational interval, when it is expected that these outputs would contain valuable information. It is worth noting that the final appearance of each contour plot slightly depends on the selection of contour levels. This may sometimes determine whether a particular fine detail is visible or not. This consideration leads to a slightly greater number of independent observations, compared to the earlier value, 536 maps, based on timestamps only. The viewers have divided the maps into five categories.

- 227 successful maps which do not contain any immediately visible defects.
- 105 maps with one or more folding defects but no other artefacts. These defects originate from contour loops with wrong sign and require parameter tuning.
- 81 maps with triangle artefacts, which are misinterpreted compass rose direction indicators. These maps will be successful once the issues with compass roses

are fixed.

- 74 maps had a reasonable general appearance, but also severe problems. Maps with both folding and triangle artefacts are listed here.
- 75 failures, which typically originate from bad input and can not be fixed.

Fig. 36 features a typical solar map with no major defects. It was produced from good quality input which was dense with gradient direction indicators. The map has several bright regions, but the contour level is suitable so that no contours are very close. The broken contours are present at the limb, where the measured intensity gradually drops from the Quiet Sun Level into the corona and background level, which is very small and well below the first contour level in the maps. The background level is set during the dimensionality reduction, so that the lowest feasible background level is selected. It is then sufficient if all the contours are visible at one point along the limb.

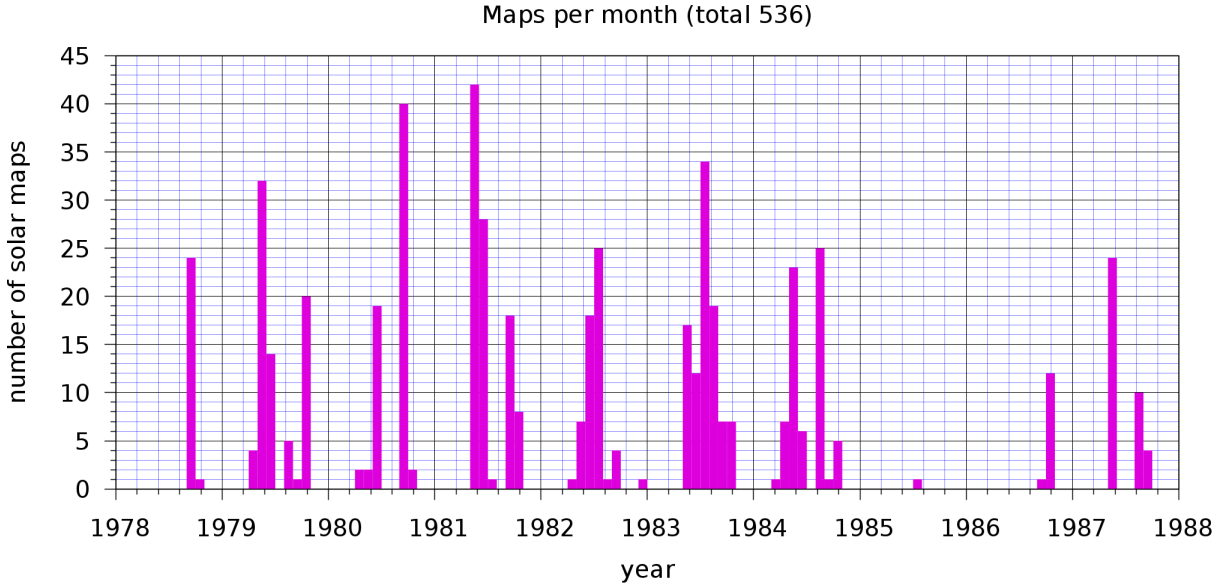

Figure 35: *Amount of maps per month. A typical observational season is from May to October. There were little observations done in* 1985 *and* 1986 *during the solar minimum.*

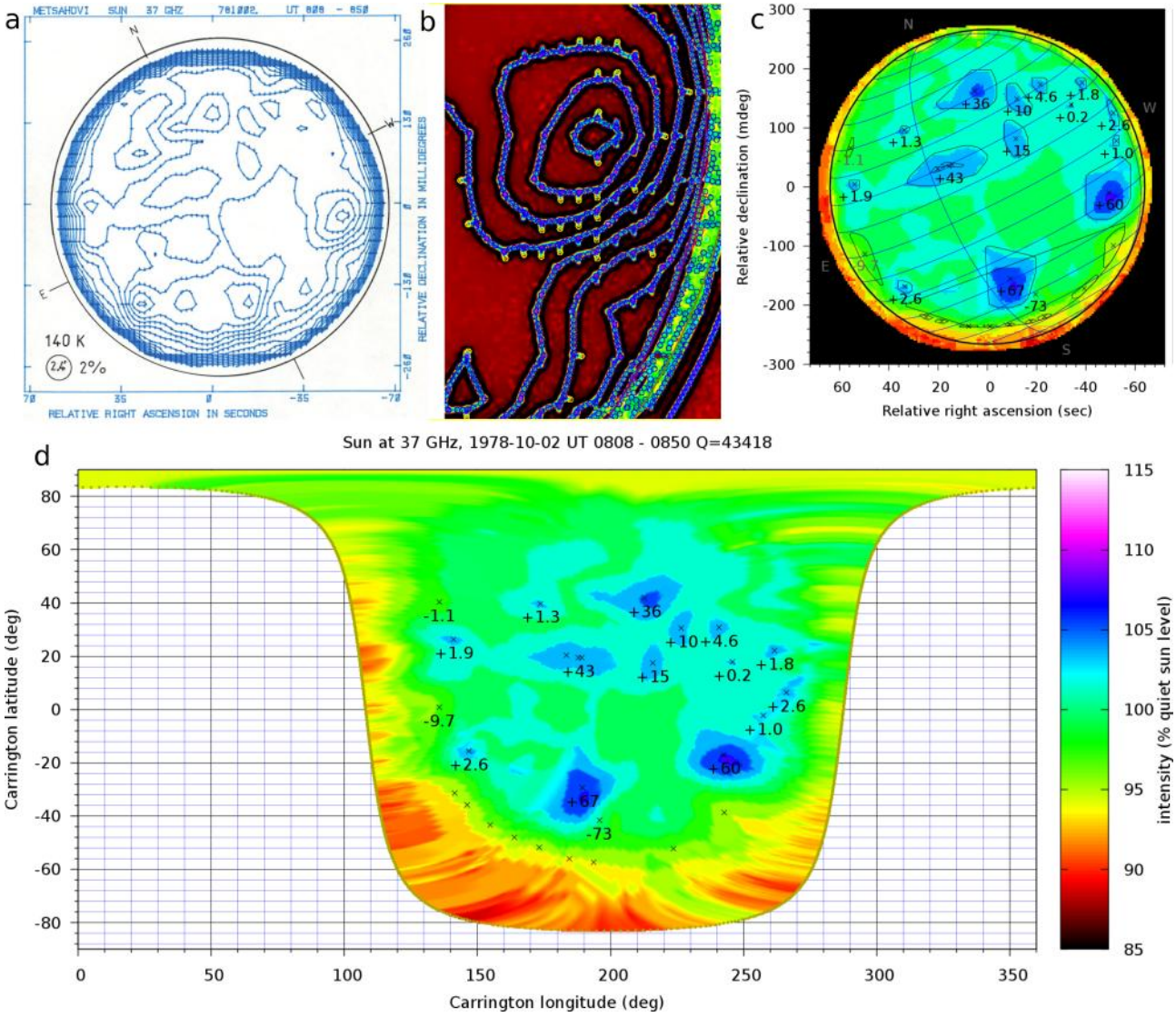

Figure 36: *a) An example of a scanned historical solar intensity map. b) Contour detection from the digital image. c) Contour plot converted to a cosine-corrected heatmap. A circle is fitted to the visual disk and an effective beam radius is subtracted from the visual radius. Some relatively bright and dim regions were identified through statistics. d) Visual intensity map transformed into Carrington coordinates. The north pole was visible from Earth at the time of the observation.*

3.1 Data coverage

The amount of observations conducted does not necessarily measure the coverage over a long time scale, since the observations may be highly sequential and therefore the solar surface has not significantly changed between consecutive observations. In order to assess the total observational coverage, a more tolerant measure has been constructed. Any equatorial feature rotates from the east and stays visible for ca. 13 days until it is lost out of sight at the west side terminator. Ideally we would want to observe any feature during this visibility window.

The solar rotational axis makes an angle of 7.25° to the ecliptic plane [55]. The rotational south pole is best visible in early March, and the north pole is best visible in early September. Approximately half of the equatorial regions can be observed from Earth at a time due to the rotation. The synodic rotation period, relative to distant stars, is defined to be 25.38 days by Carrington [56] and based

on average rotation period observed from sunspots. Since the Sun has no rigid surface and the convective zone follows differential rotation, this definition is still very arbitrary. It is, however, useful to map solar events to this rotating coordinate system, the Carrington coordinates. An event is seen to travel west on the visual solar disk (Fig. 37). When plotted on Carrington coordinates, the event remains relatively stationary (Fig. 38), as the rotating Carrington surface tries to cancel the average rotation of the Sun.

The zero meridian in these heliographic coordinates is defined in terms of ascending node. The intersection of the solar equator and the zero meridian crossed the Earth's orbital plane at noon on January 1, 1854, in Greenwich time. The first Carrington rotation begins late on November 9, 1853, which is when the zero meridian was incident with the central meridian seen from Earth. The central longitude decreases from 360° to 0 during each Carrington rotation. [56]

The zero meridian always separates the solar surface into two sections, out of which the one to the west is the new Carrington rotation, and to the east is the old rotation. The meridian opposite to the central meridian is another divider. Regions to the west are in the old rotation. Following this logic, observations are binned into separate Carrington rotations, so that we can measure how well the historical solar surface is covered by Metsähovi data. The statistics are shown in Fig. 39. This does not yet include those 37 maps for which the layout failed.

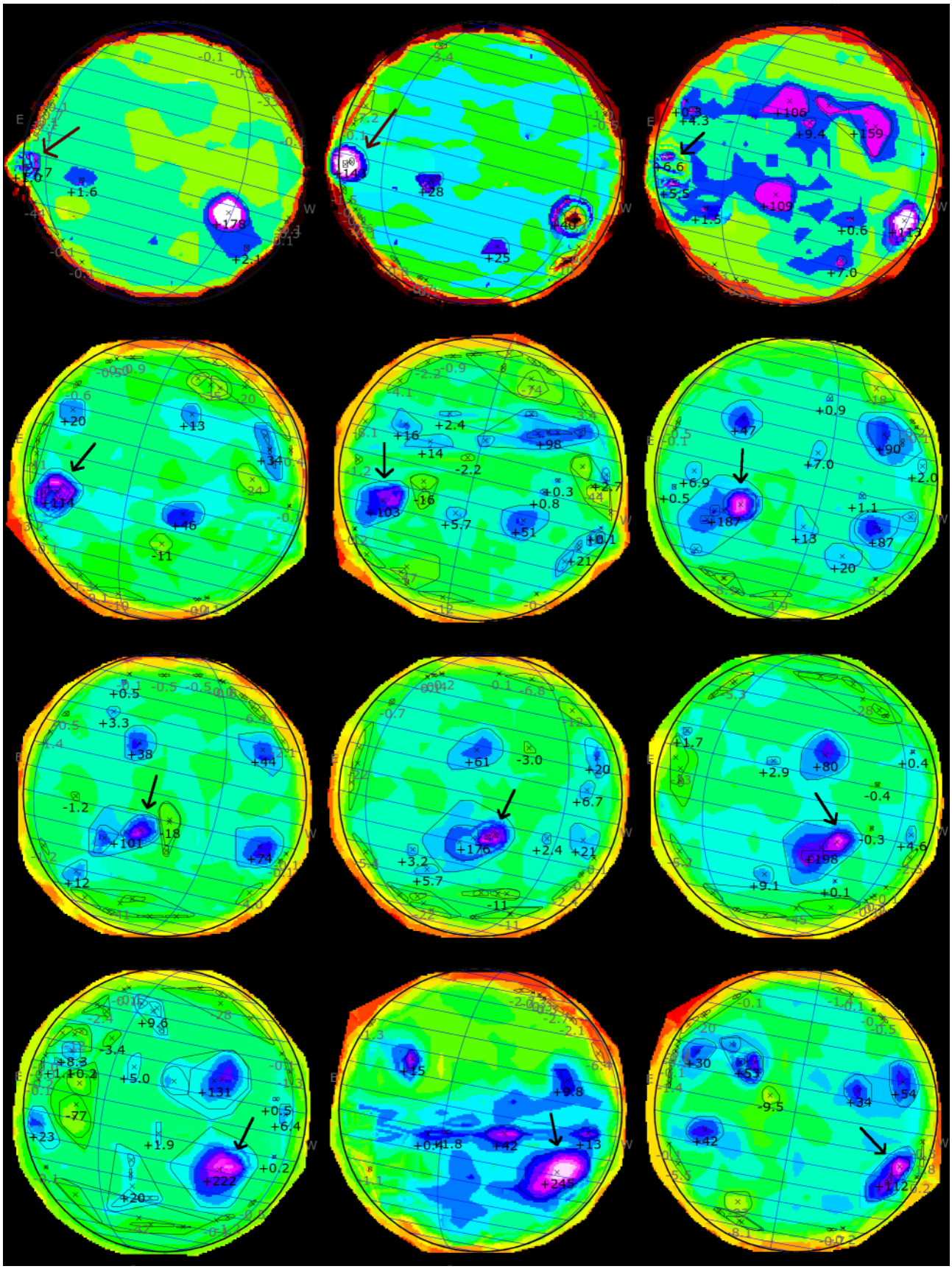

Figure 37: *Bright region, marked with an arrow, travels westward on visual solar disk. Daily observations on June* 2..13, *year* 1980.

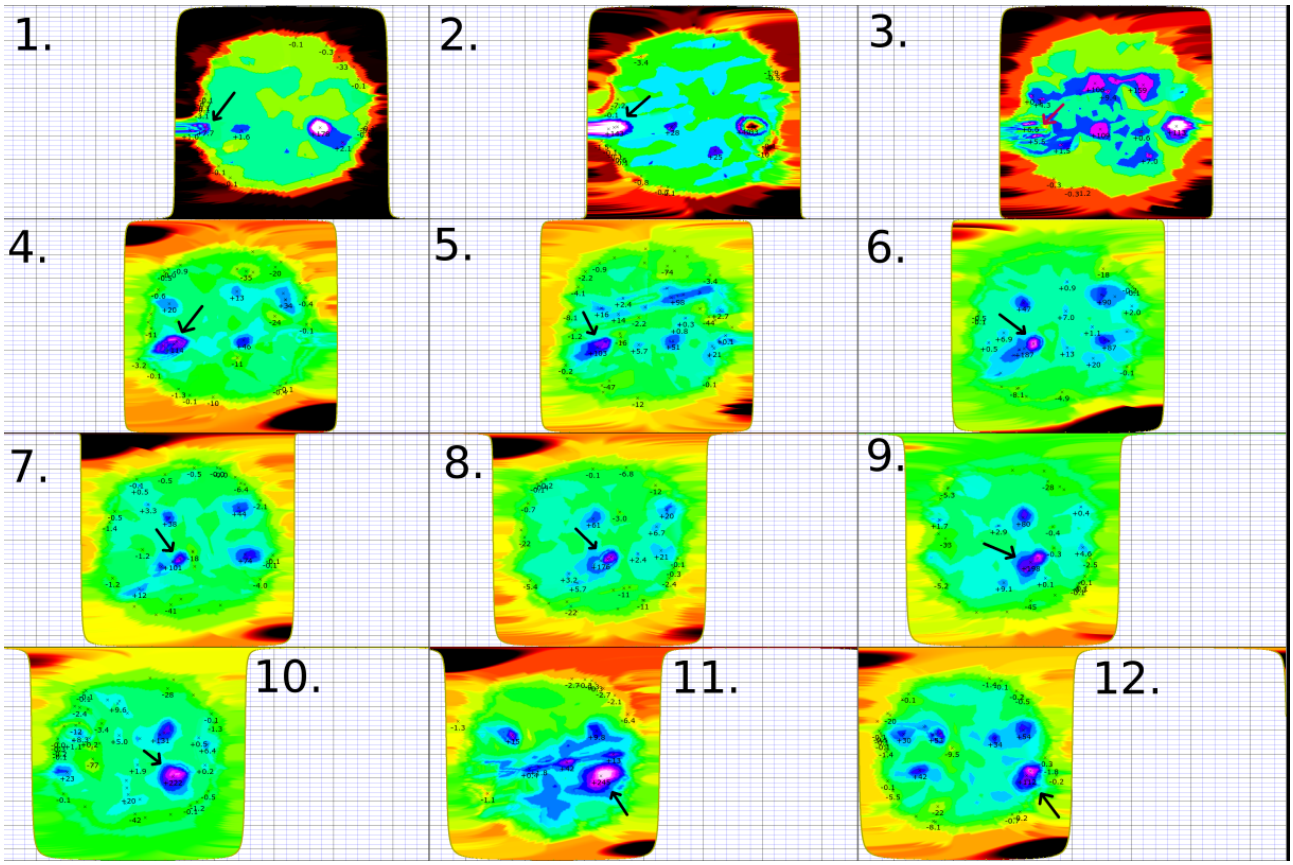

Figure 38: *Bright region, marked with an arrow, on heliographic coordinates. Daily observations on June 2-13, 1980. Data is only available from the hemisphere which is visible from Earth. This hemisphere travels east as the Carrington surface rotates relative to the observer.*

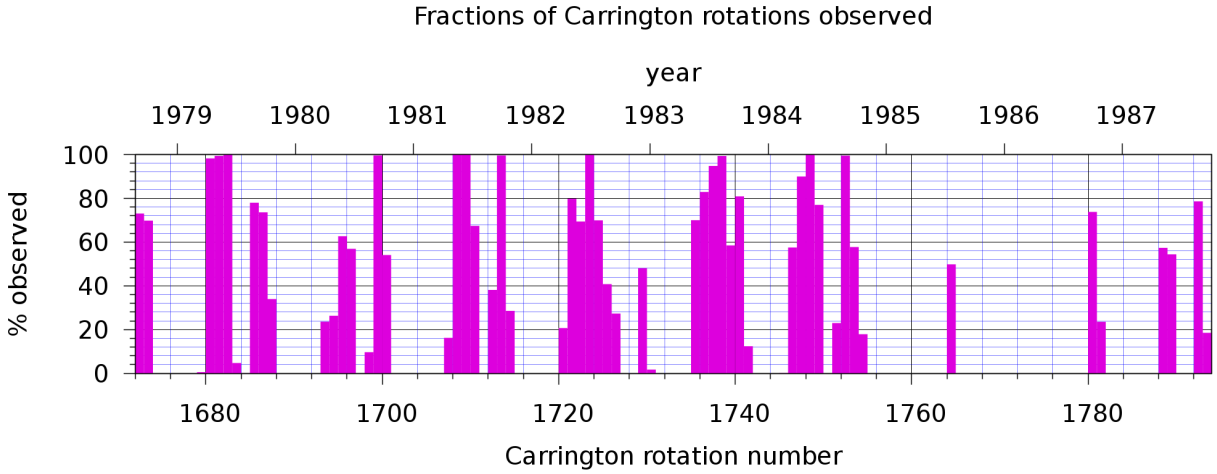

Figure 39: *Fraction of solar surface observed during each Carrington rotation. Each rotation begins when the solar zero meridian coincides with the central meridian seen from Earth. Longitudes from* $360°$ *to* $0°$ *are included in the rotation, during which they coincide with the central meridian in that order.*

3.2 Bright and dim regions

When performing regular observations, a single bright or dim region is observed multiple times during its ca. two weeks course across the visible solar disk. In order to assess the output of the *sunmap* program, an additional ad-hoc program *follow*, accompanied with a script *follow.sh*, was written as a part of this thesis. The program parses the JSON output of the *sunmap* program, and links repeated regional observations into a single track, based on spatial and temporal proximity:

$$d = \sqrt{\left(\frac{\Delta L}{1^\circ}\right)^2 + \left(\frac{\Delta t}{3\,\mathrm{d}}\right)^2}, \tag{171}$$

where the distance is in degrees on the Carrington surface, and Δt is the time passed between consecutive observations. For track continuity, we require that $d \leq 15$. When more than one candidate for the continuation is available, we choose the one with smallest d.

We somehow need to process the multiple renderings of same map. Here, an average brightness is calculated when there is brightness information available from multiple renderings. We require that the tracked region has to be observed at least 3 times on unique observational data sets, and the observations have to extend over two days. This approach produces 791 tracked regions and leaves 3936 sporadic observations from the data set of ten years. The tracked regions were further divided into 516 stationary and 275 migrating regions, based on whether the region migrated more than 15° on the Carrington surface. Sporadic regions are observed in only one map. Their absense in other maps may result from defects, or the data may be sparse at the time of observation. As future work, also the absense of a bright or dim region should be treated as information. All the observed regions are plotted in Fig. 40.

The stationary regions are observed to pass from east to west as the Sun rotates. When we compare their Carrington longitude to the central meridian as seen from Earth, we get the visual longitude, which is zero when the region is seen at the centre of the visible solar disk. Suppose that the all the observations are performed at random accasions and any region is always detected at random stage of passing. Then we would expect the distribution for visual longitude to be uniform when we are clearly away from them solar limb. Another possibility is that the observers are actively searching for bright regions, so that once detected the regions are regularly observed until they are no longer visible behind the limb in the West. With this practise, we would obtain a steadily increasing slope in the distribution for visual longitude.

In Fig. 41 and 42, however, no significant slope is observed. This suggests that the intensity maps were collected at random occasions. There possibly is a slight decreasing slope with the dim regions. This could be explained if the dim regions

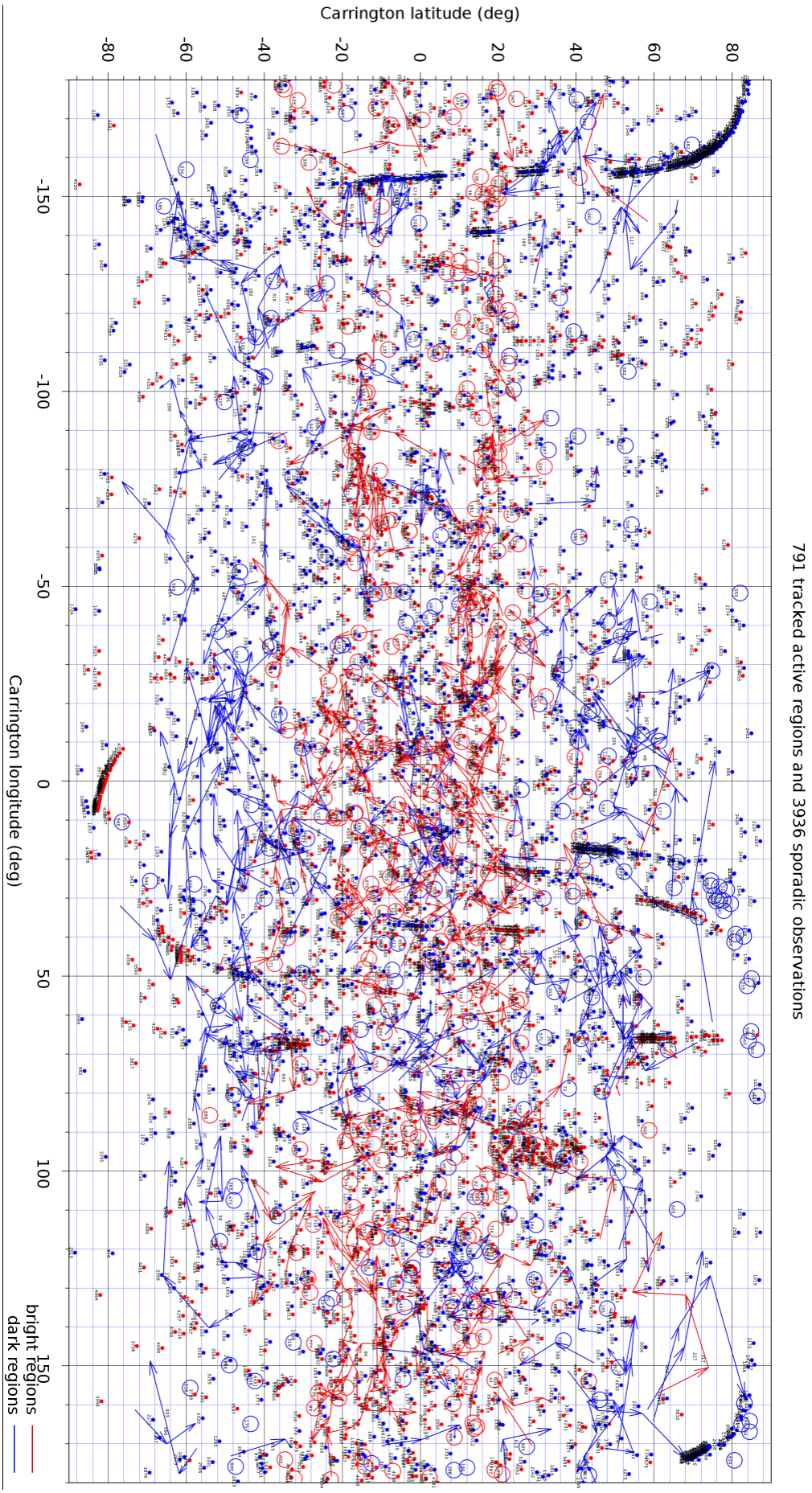

Figure 40: All active regions observed. Sporadic regions (dots) are observed only once, whereas stationary (circles) and drifting regions (arrows) are seen multiple times which cover several days. Red areas are brigher, and blue areas are dimmer, compared to Quiet Sun Level.

are related to temporarily low solar activity and therefore reduced interest in performing solar observations. However, no solid assumptions can be made until all the defects related to dim regions are fixed. In Fig. 41 and 42, we observe limb defects as linear patterns of dim regions near the poles and high absolute central longitudes.

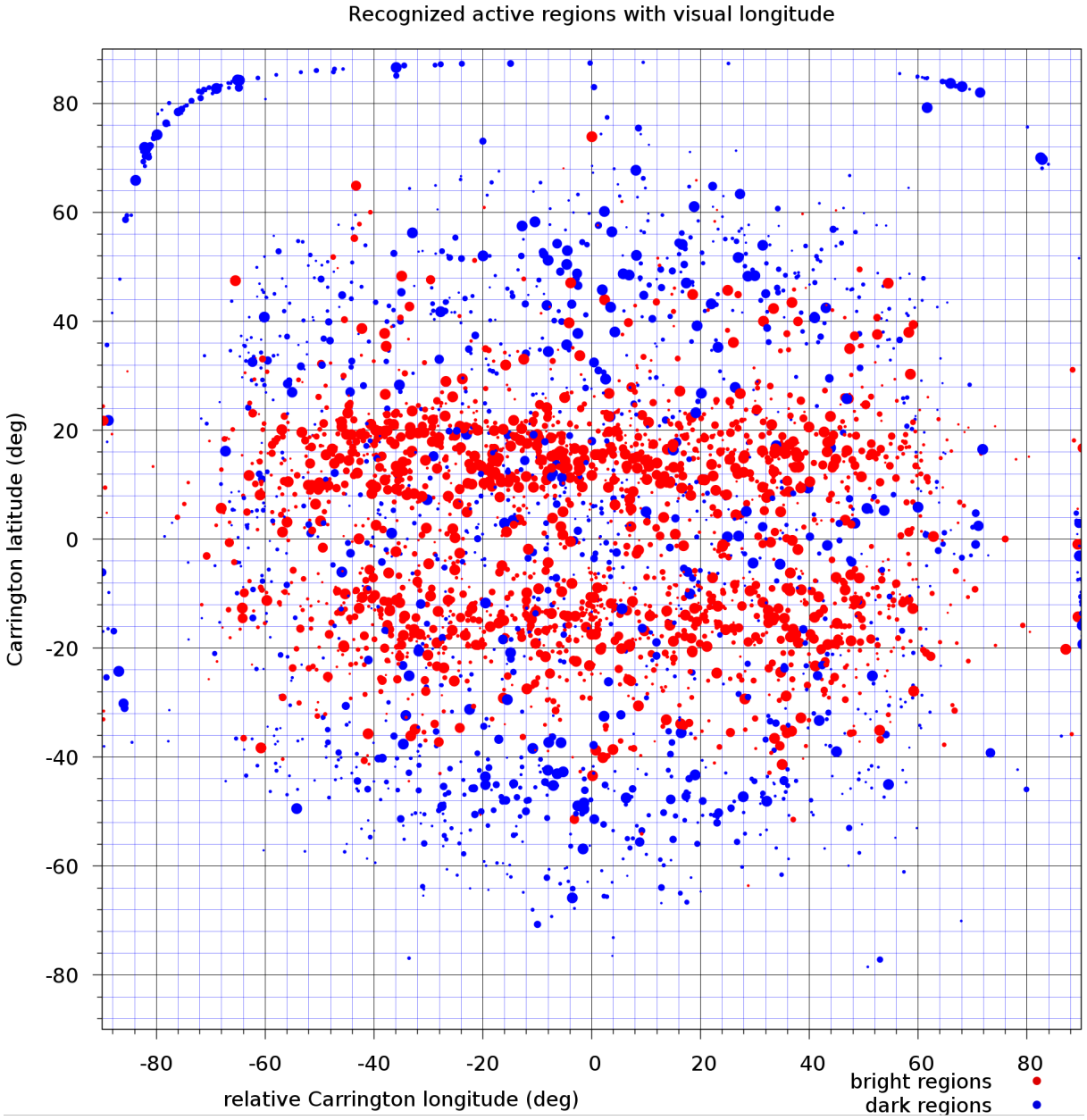

Figure 41: *Distribution of active regions plotted with respect to the central meridian seen from Earth. The total power, relative to Quiet Sun Level, is indicated with the size of each red or blue circle. The pattern is fairly symmetric. There appears no purposeful attempt to keep on observing each spotted active region until it is lost behind the west-side terminator due to solar rotation. These attempts would show concentration of observations on the right, and no such bias is seen here.*

When the bright regions are plotted for solar latitude, we observe two peaks at the middle latitudes as in Fig. 43. The dim regions are concentrated at the equator and at high latitudes. The bright regions are rare at the high latitudes. We observe dim regions at the south pole, which is visible to Earth after early June. Very few dim regions are observed at the north pole, which is visible until early

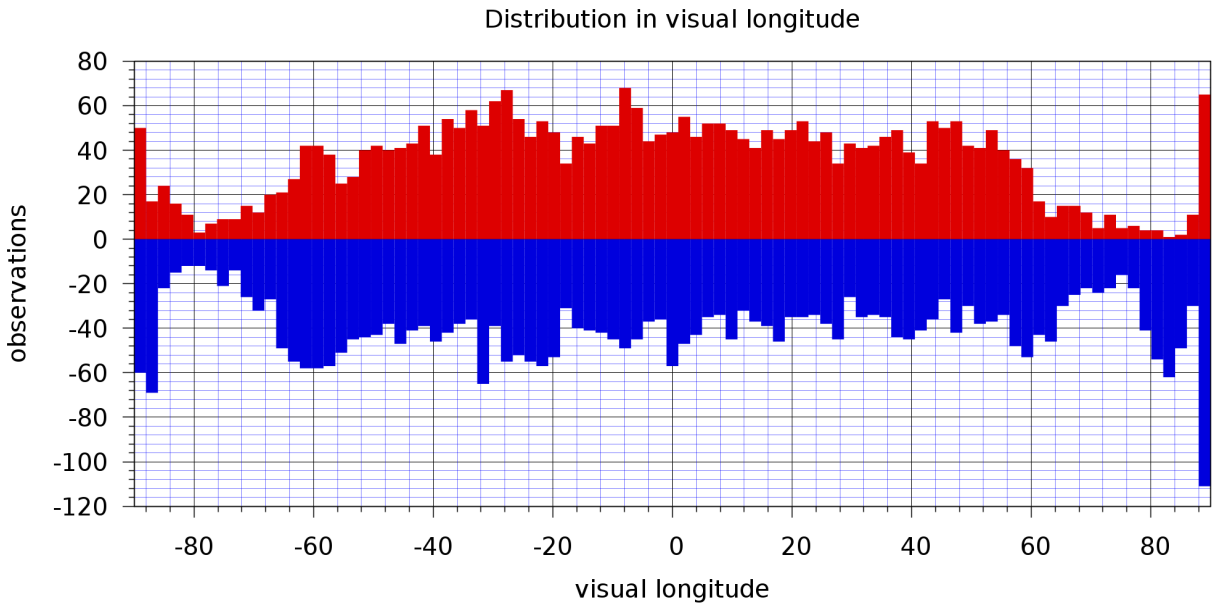

Figure 42: *Amount of bright (red) and dim (blue) regions vs. the visual longitude.*

June. The observational season is from May to October, so this is reasonable. In order to have better statistics in the future, we need to take into account the observational coverage of different latitudes.

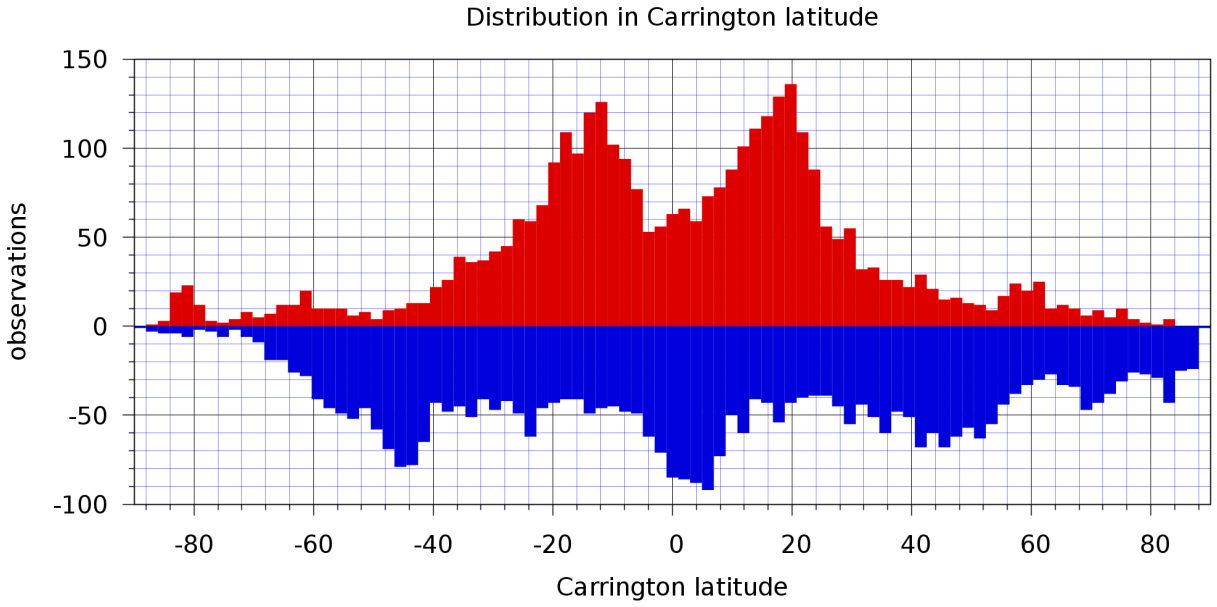

Figure 43: *Solar latitude distribution of bright and dim regions in the* 10 *year contour dataset.*

The bright regions appear at the high latitudes at the beginning of the cycle, and later they appear closer to the equator. This can be observed in Fig. 44, which is similar to the famous sunspot observations shown in Fig 7. With the dim regions, no clear pattern is observed (Fig. 45), and better analysis is needed. We need statistical tests to treat varying observational activity during the solar cycle, which is future work.

The absolute power of a bright or dim region can be calculated as relative to the Quiet Sun Level. This method is still unreliable, since many maps have unrealistic

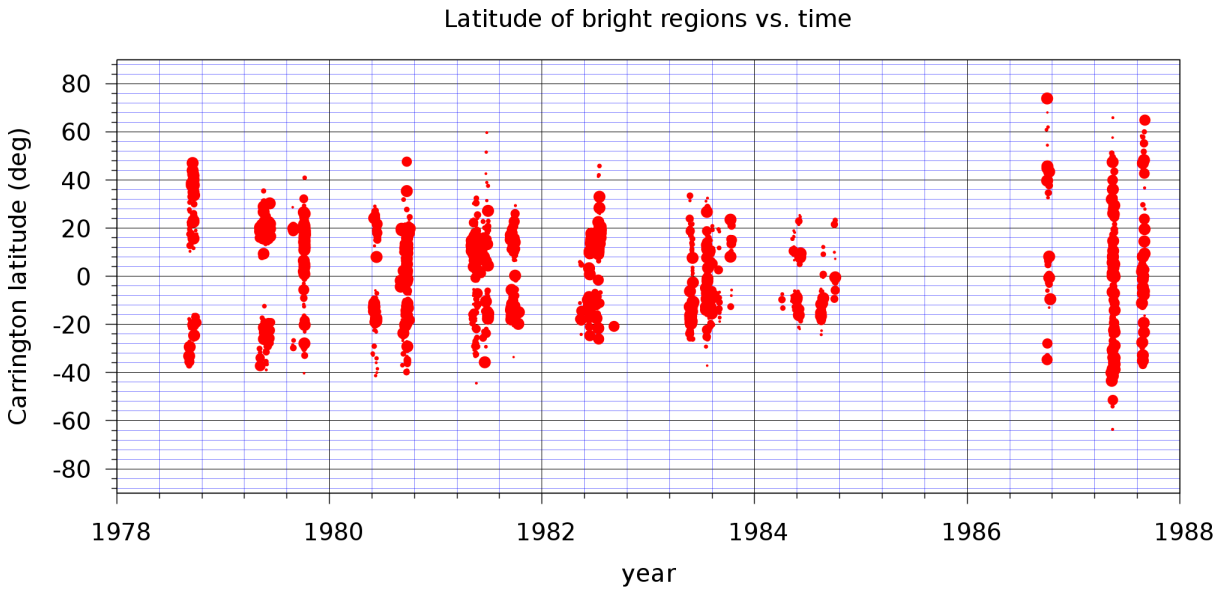

Figure 44: *Observed bright regions at various latitudes vs. time. The solar cycle changes during the quiet years* 1985*and* 1986*, where very few observations were made.*

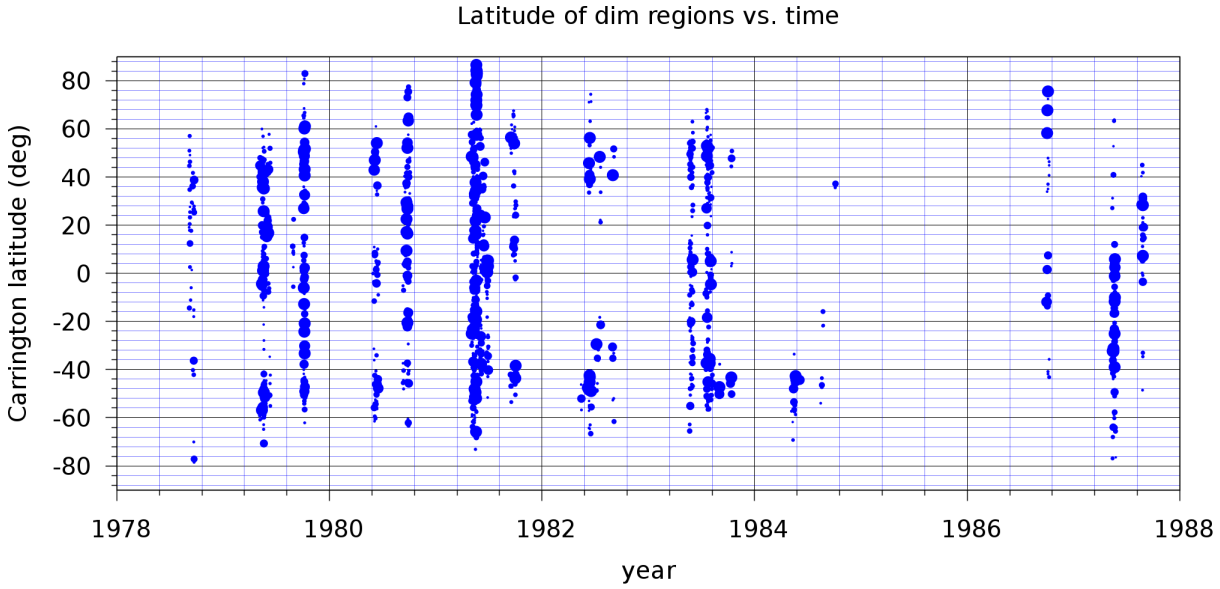

Figure 45: *Observed dim regions at various latitudes vs. time.*

contour level scaling. However, most of the regions have low power. The power per surface area must deviate more than 2σ from QSL, so that once integrated over the region of area A, the the absolute power will always exceed $2\sigma A$. We assign a negative power rating for dim regions. For areas of very low absolute power, dim regions dominate, but may be explained by limb darkening effects (Fig. 46). In the future, we will study also the area distribution and the fine details which is also supplied by the code. Any observed regions may contain several extrema, but here only the total power is counted.

When we are close to the limb of the solar disk, the exact placing on the Carrington surface is prone to projection effects. We thus neglect all regions that are observed more than 80° degrees away from the central meridian, or more than 50° away from solar equator. Having so filtered the data, we can study the drifting of the regions. Drifting along a circle of latitude would signal differential rotation, while

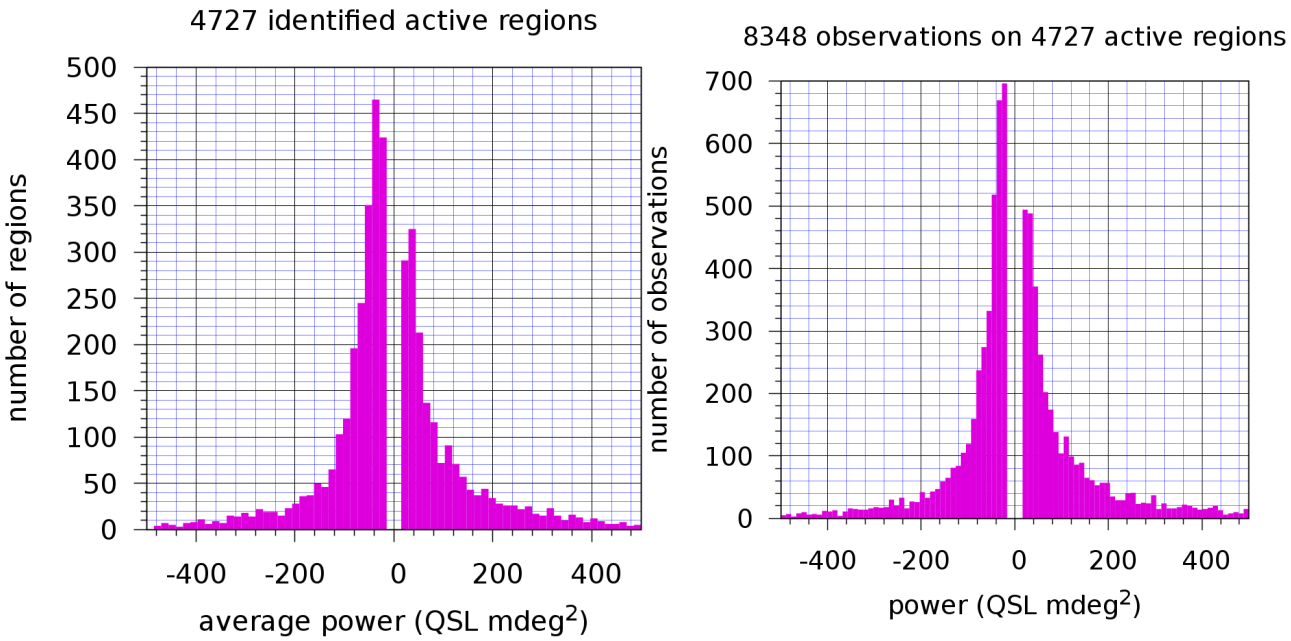

Figure 46: Power distribution of bright and dim regions. Very weak regions were neglected. a) Each track is counted only once. b) Each observation is counted even if the region is re-observed later.

drifting towards either pole constitute the meridional flow. Neglect tracks which drift more than 3 degrees per day we and obtain Fig. 47 for differential rotation and Fig. 48 for meridional flow. In the future, these data sets will benefit from further statistical analysis.

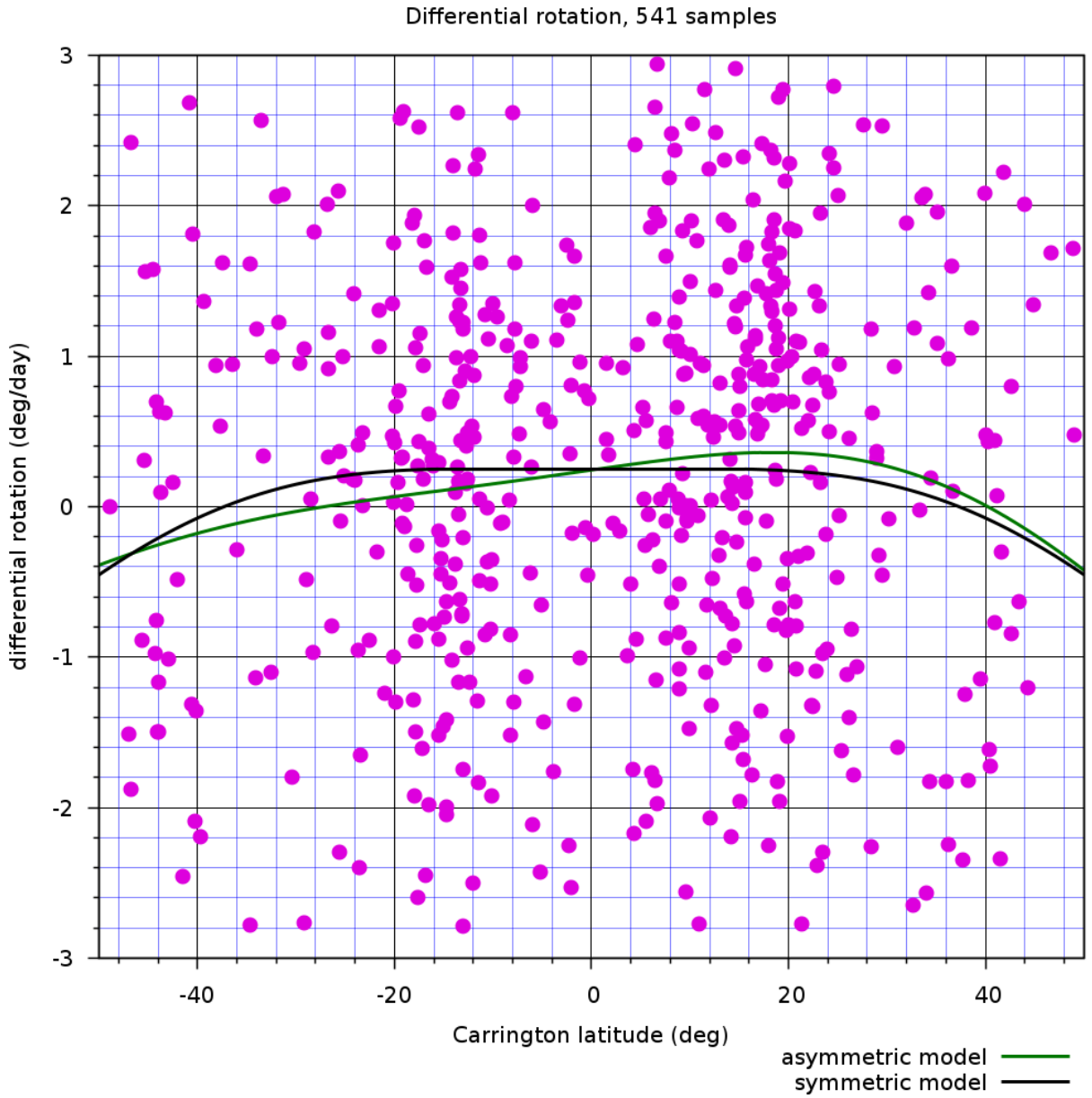

Figure 47: *Differential rotation distribution of the bright and dim regions.*

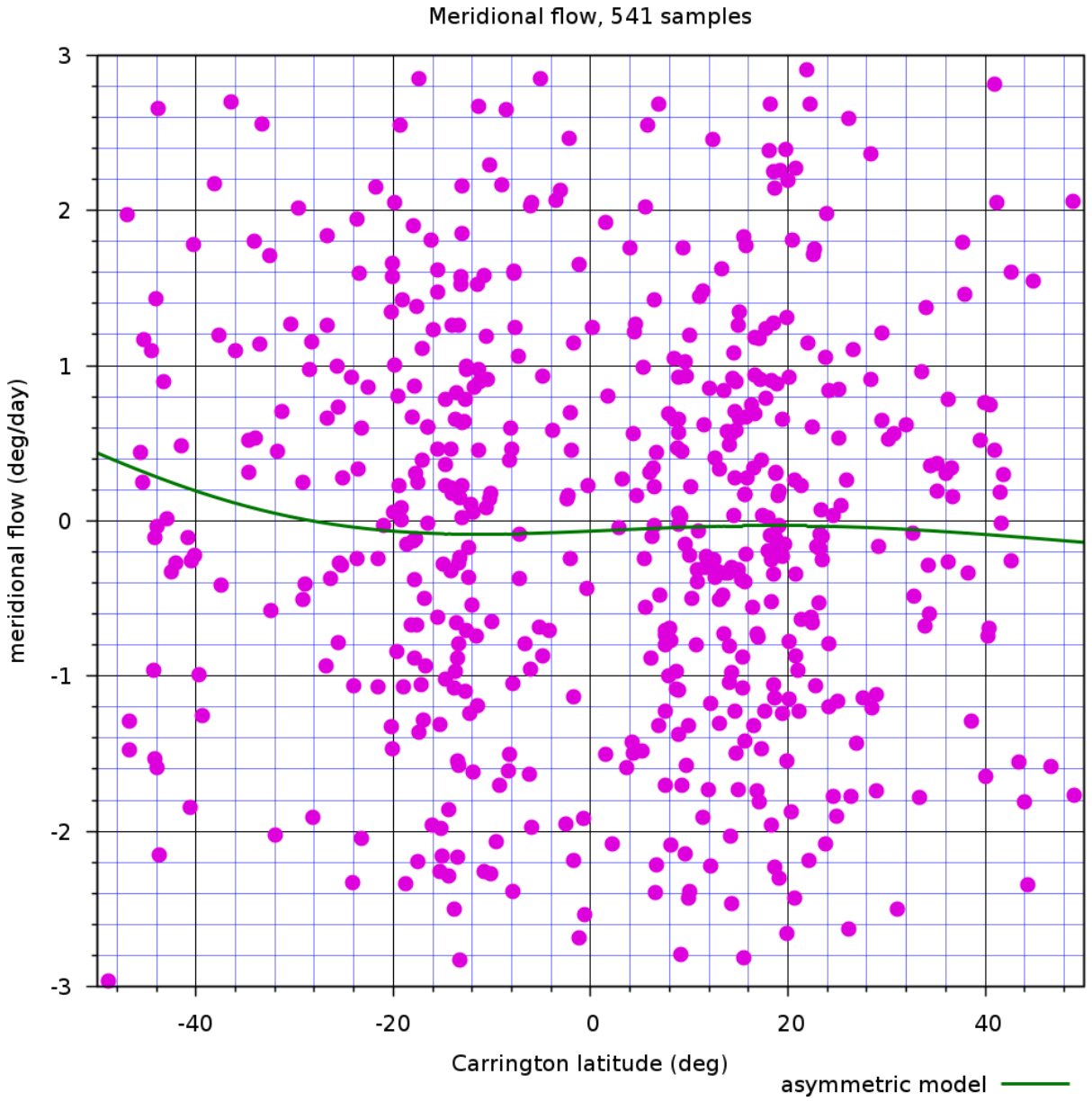

Figure 48: *Meridional flow distribution of the bright and dim regions.*

3.3 Output quality

There are maps for which, although the algorithm works well for the most part and good results can be obtained, the code produces some disturbing artefacts in the output (e.g. Fig. 49). Some of the contours were left broken in the original plotting process, and were misinterpreted in digitisation. The output intensity profile includes obvious inaccuracies. The Poisson solver produces gradients in locations where the potential field should be locally constant. The dimensionality reduction loop in the Poisson solver fails to eliminate the perturbation when it is supplied with conflicting contour information.

If we fail to interpret some layout features, they may appear as bogus contours. For example, there are two numbers near the circular antenna beam size indicator at the bottom left corner in Fig. 49. The contour level spacing is given both as signal temperature 185 K and as a percentage of the Quiet Sun Level (2.5%). These labels are tried with two different stencil typefaces, and the one with better quality is chosen. Unfortunately, an unsuitable typeface has been selected, despite its failure to interpret the label 2.5%. This could be easily fixed with an approriate modification in the selection logic. The process of writing the software has involved several similar steps where small hacks have gradually improved the output quality. However, a more comprehensive method hopefully solves the text recognition problems in the future.

The present algorithm tries to determine the solar terminator as a convex hull of contours. The convex hull is separated into straight and curved sections. The straight sections are usually related to data cropping, while curved sections belong to the solar disk terminator. A circle is fitted to the curved sections, and if individual points deviate from this circle by a predefined relative threshold, they are excluded.

With the example in Fig. 49, the label 2.5%, is now interpreted as a contour, and it is too close to the solar disk terminator, so that it is considered as part of the solar disk. A larger threshold would eliminate the defect, but it might also cause additional problems with other maps. Again, by adding complexity and slightly extending the logic we could possibly handle this particular case. One option would be to try all suitable combinations for the solar disk. Three points are required to define a circle. When the convex hull has n vertices, there are $\binom{n}{3}$ subsets, each of which is a potential substrate for the terminator circle. It is reasonable to try the circle fitting for all of these combinations, or maybe even for $\binom{n}{4}$ subsets with one extra vertex for each. We can then choose the candidate which produces a circle with the least standard deviation with the other hull vertices. We could the have much smaller error threshold and more reliable detection of the solar disk, and therefore less sensitivity for defects outside the disk.

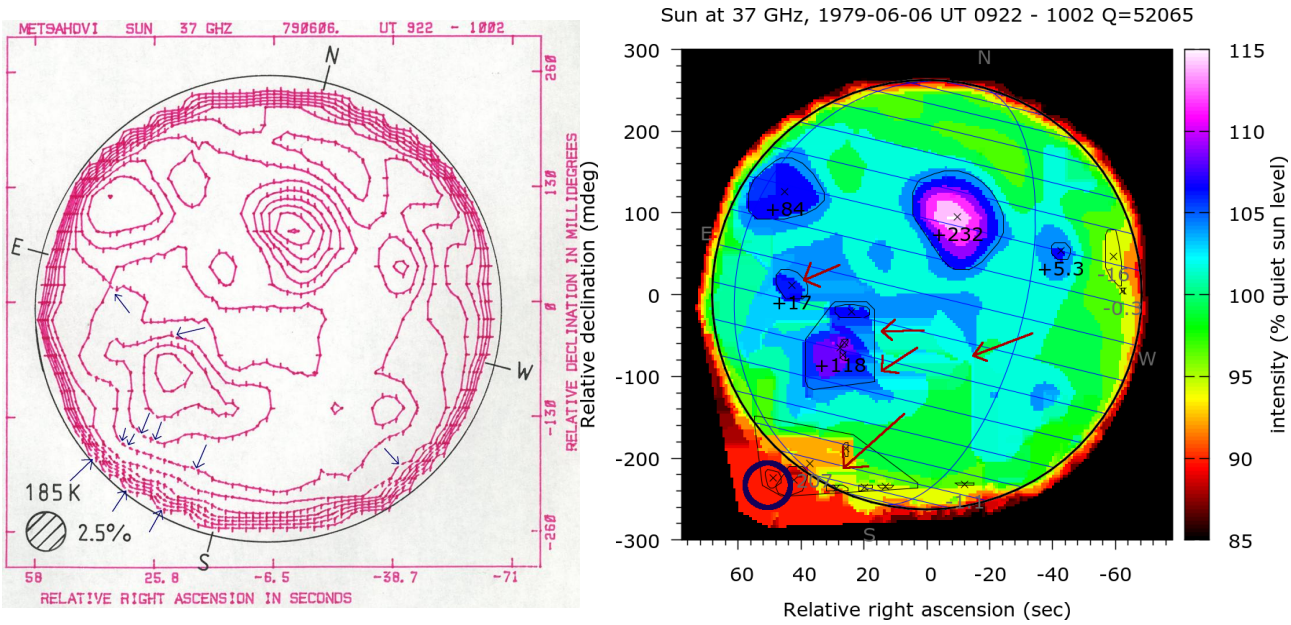

Figure 49: Mostly successful map with some troubled contours marked with red arrows. An invalid solar disk terminator is identified with a dark blue circle.

Another common failure mechanism is shown in Fig. 50. The compass rose is not detected, and it will instead be interpreted as contours. The underlying cause is that the program tests first for four separate lines and then for two crossed lines. In Fig. 50, the N-S -axis is complete line, while W-E consists of two separate lines. The hand-drawn curve representing the equator is not a typical feature in the scanned maps, and thus it is not feasible to develop code for detecting it. However, we could consider a system for detecting and eliminating stray lines from the plots. Any error which causes the compass rose go undetected will result in similar pattern in the output.

The contour direction has to be correctly determined. In Fig. 51, some of the gradient direction indicators are misinterpreted, resulting in a folding pattern instead of a single intensity maximum. Suppose the brightening should be n contour levels high, it will end up being $n-2$ levels instead, when one contour is interpreted with wrong sign. In Fig. 51, there is another defect near the terminator, where the brightening has broken contours. There defects can be fixed by tuning the detection parameters for gradient indicators and contour lines.

The current line detection is sensitive to total contour mass. If there are too few contours, shadow and speckle noise will be interpreted as circular markers. With a sufficient concentration of false circular markers, there will eventually be intact contours which cause defects in the map (Fig. 52). This problem will be fixed when the brightness threshold is set automatically based on the level of local variations in image brightness. Once this adaptive contrast is implemented, it will be a boolean argument to be given from the command line.

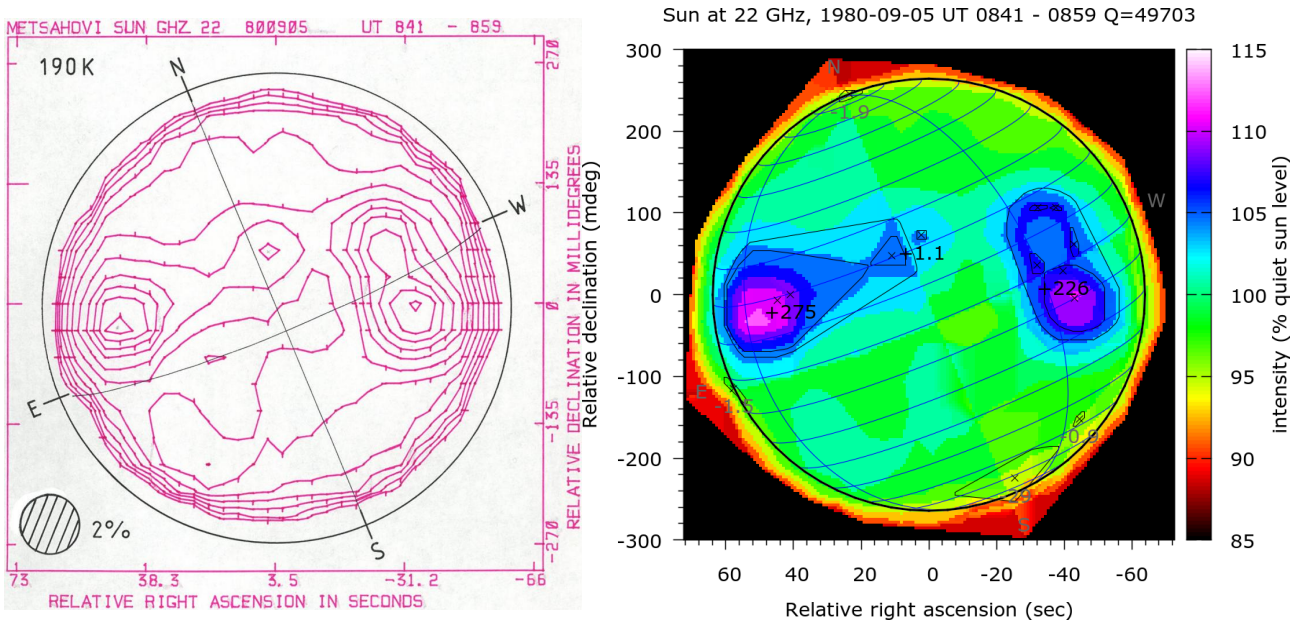

Figure 50: Atypically marked compass directions are not properly detected with present settings. As a result, the letters N, S, and E as well as the lines associated with the directions are interpreted as contours. This causes a cross pattern on top of the resulting intensity map.

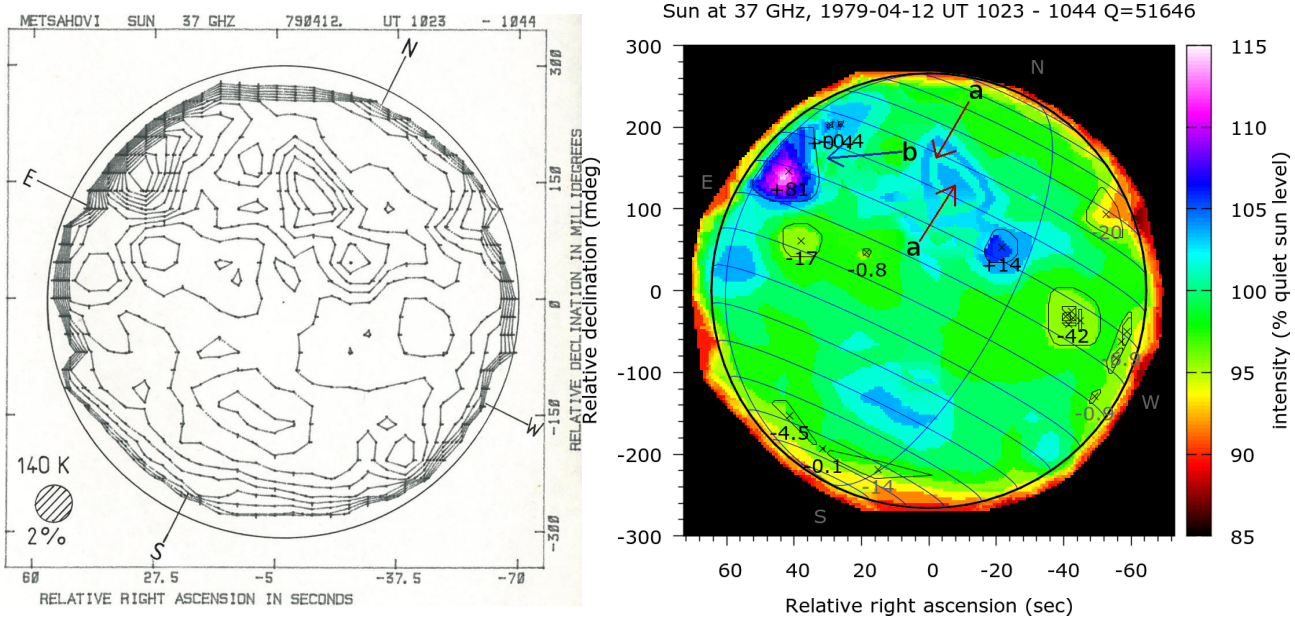

Figure 51: Folding pattern instead of a single intensity maximum arises from one contour having wrong sign (red arrows). There is brightening near the rim with broken contours (blue arrow). Due to this defect, the brightening appears to have several maxima.

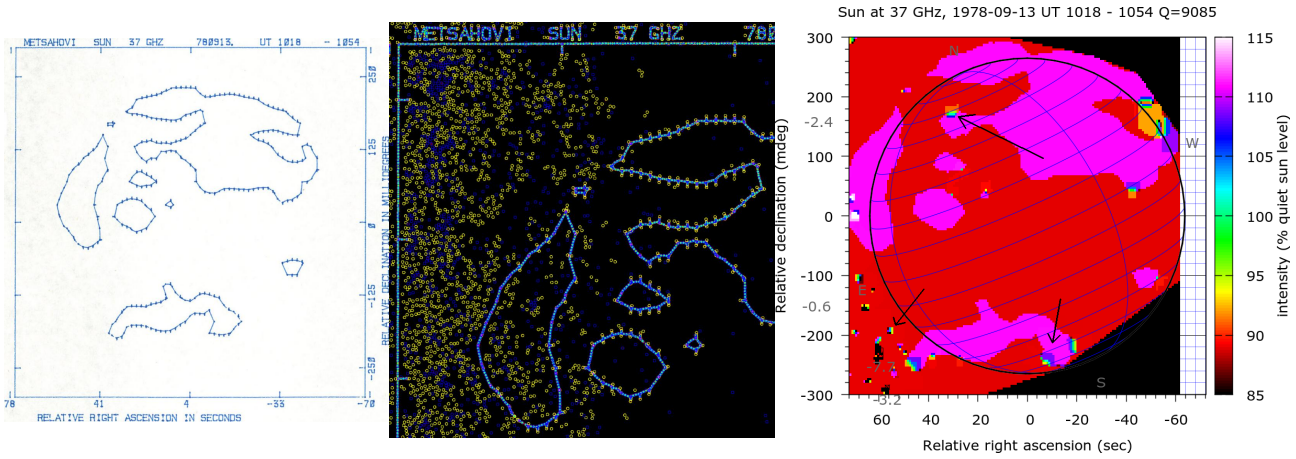

Figure 52: This map contains very few contours, even though a constant histogram fraction of ca. 0.08 is pre-defined to contain contours. As a consequence, bogus contours are observed at locations marked by black arrows.

In Fig. 53, we observe a typical level defect where one of the long terminator contours is interpreted with wrong sign. This results in an elongated brighter feature at the rim. In Fig. 53, a series of very small dim regions are detected at the rim near the south pole. It is usually not feasible to place dim regions at the limb, where we can expect a lower overall intensity than the QSL. For example, the 37 GHz radiation originates from the chromosphere [57], and beyond the chromosphere the measured intensity is below the first visible contour line in the historical maps. The beam size of the antenna limits the feature size on the map, and thus the intensity has to drop from QSL into almost zero within an angle which is comparable to the beam diameter. Thus we would expect to see low intensity at the limb, and we should not misinterpret that into dim regions. However, it is possible that there actually is a dim region near the terminator. In order to find better balance in detecting dim regions near terminator and avoiding false positives, the cutoff distance to the disk perimeter can be adjusted with the command line parameter *lobe_shrinkage* (Appendix. B.15). An improvement for the present software would be to search for the true solar intensity distribution in a parametrised form. As we know the antenna convolution and the measured intensity map, we can search for such a true intensity distribution which closely resembles the observed map after antenna convolution has been applied.

In Fig. 54, the original data contains a tail of high intensity extending southwest on the visible solar disk. The algorithm cuts off all the contours that reside beyond a threshold distance from the compass rose, and everything outside that threshold is set to zero contour level, which represent black space. Thus several perimeter contours are broken, which supplies conflicting information to the Poisson solver, that is observed as defects in the output.

An example of a totally failed map is in Fig. 55. The coordinates for y axis are obviously nonsense, but for x coordinate they suggest that the Sun should be at the centre and completely visible. For this reason, most contours are cut where the perimeter should be, and the black space is assumed outside. The code is also able to detect the solar perimeter from the contours, but this information has lower priority since the coordinates are partially successful. Obviously this particular map can be fixed if we increase the priority for contour based coordinates.

The Poisson solver should be able to handle conflicts that arise from broken contours at the perimeter. When the conflicted area is too large, dimensionality reduction does not produce a consistent result anymore. This is observed in Fig. 56. Some of the maps have wrong contour level spacing marked on the input. For these maps, the intensity scaling has to be set manually. An example of such a map is given in (Fig. 57). When the radio brightening occurs at the rim of the visible solar disk, often a thick bundle of contours occur (Fig. 58). With current status of the code, overlapping gradient directions indicators may end up being

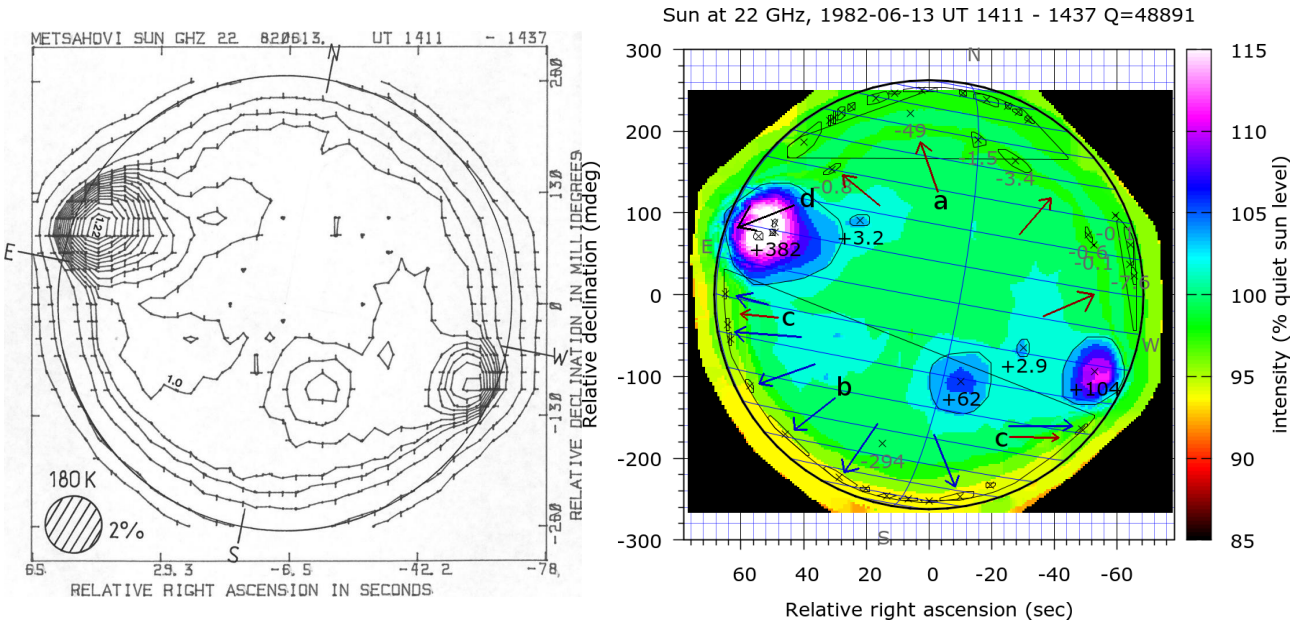

Figure 53: Red arrows (a): long folding defect originates from misinterpreted contour signs. Blue arrows (b): bogus dim regions which arise from the limb. Red arrows (c): Path detection produced broken contours. Black arrow (d): broken contours around an intensity maximum result in a shadow defect.

associated with a wrong contour line, resulting in the adjacent contour to have opposite sign and a folding artefact. We could benefit from further improvements when it comes to detecting gradient indicators, as can be seen in Fig. 59. It is possible that the map layout can not be detected, in which case no results are produced. There are 47 such out of 1012 contour-plots. For Fig. 60, the original map was plotted such that the labels for the dec axis are visible. For Fig. 61, the ink-drawn lines are too weak and thus the box lines are broken. For this case, the box detection tolerances can be altered or the map can be manually retouched. If the box lines have gaps, the parameter *line_extend_dist* can be altered to allow the line constructor to skip larger distances. Both Fig. 60 and Fig. 61 belong to sets of three scanned maps contining the same information, so for these particular maps no information was lost.

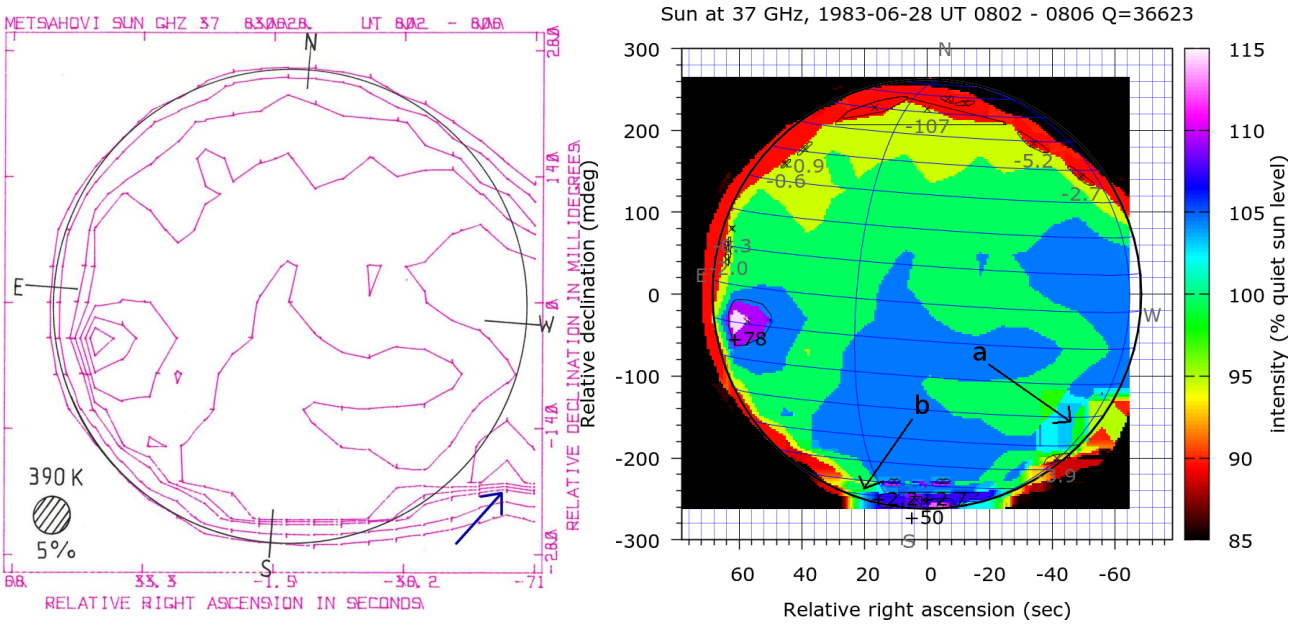

Figure 54: a) Bogus tail of high intensity is cut off, but conflicting information remains. b) Broken contours produce an intensity gradient in the output.

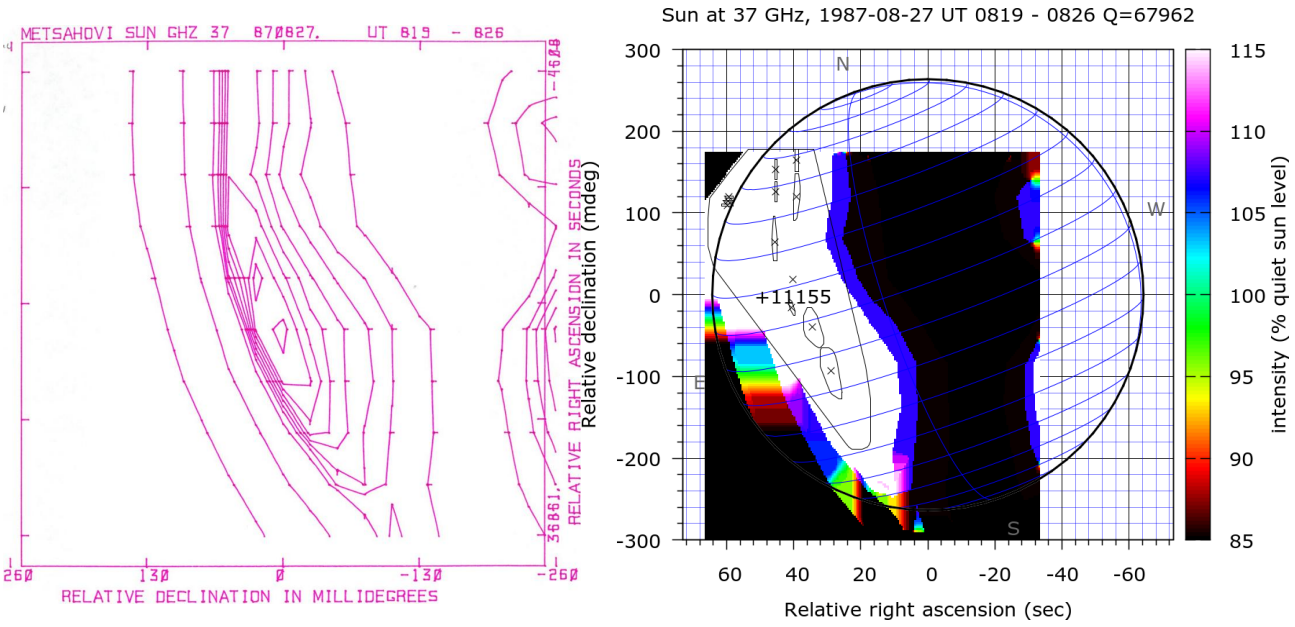

Figure 55: The original map has unfeasible coordinates for the y axis, so as a fallback, the Sun is assumed to be centred. This leads to broken contours, even though this can be fixed by altering the priorities in the decision making.

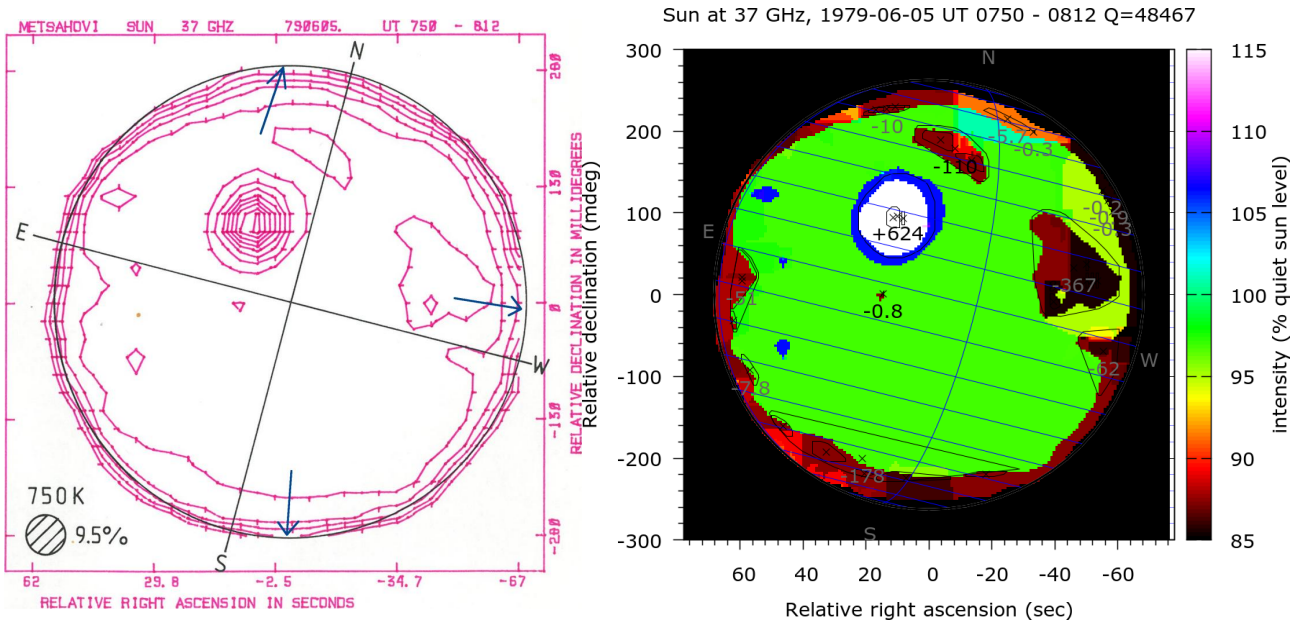

Figure 56: The map should be cropped at the three sides marked by arrows. This is not done, and the conflicting information is supplied to the Poisson solver.

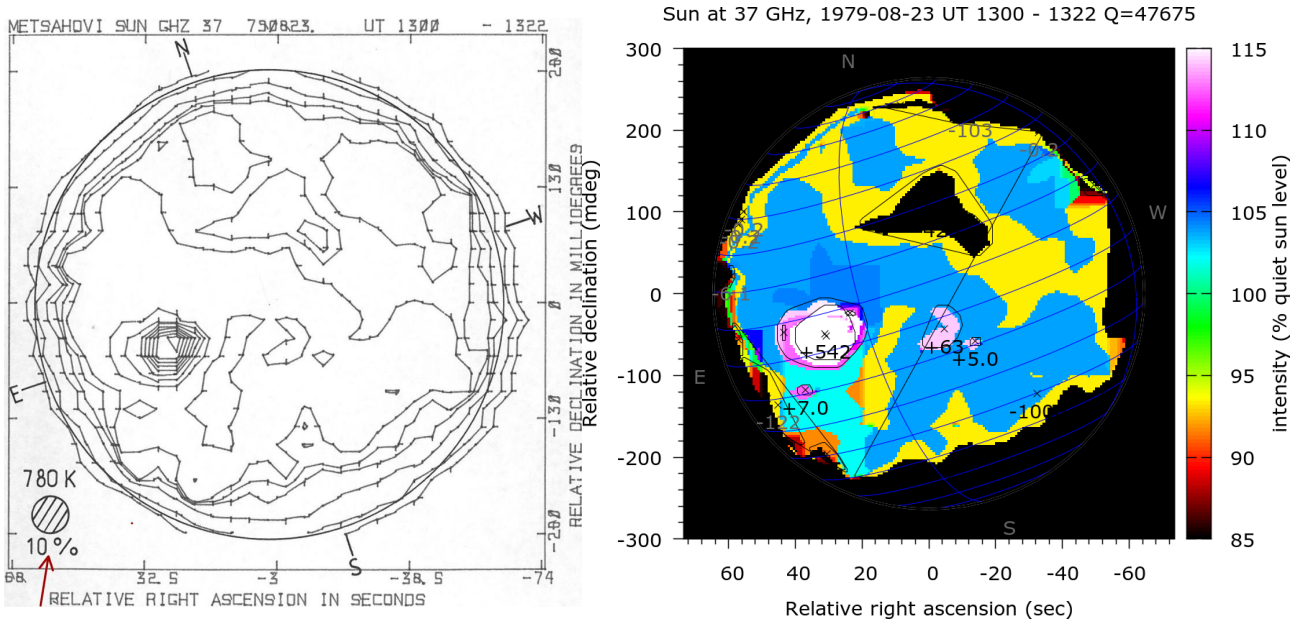

Figure 57: The input has stencil marked contour level spacing 10%, which would mean unrealistic variation in intensity.

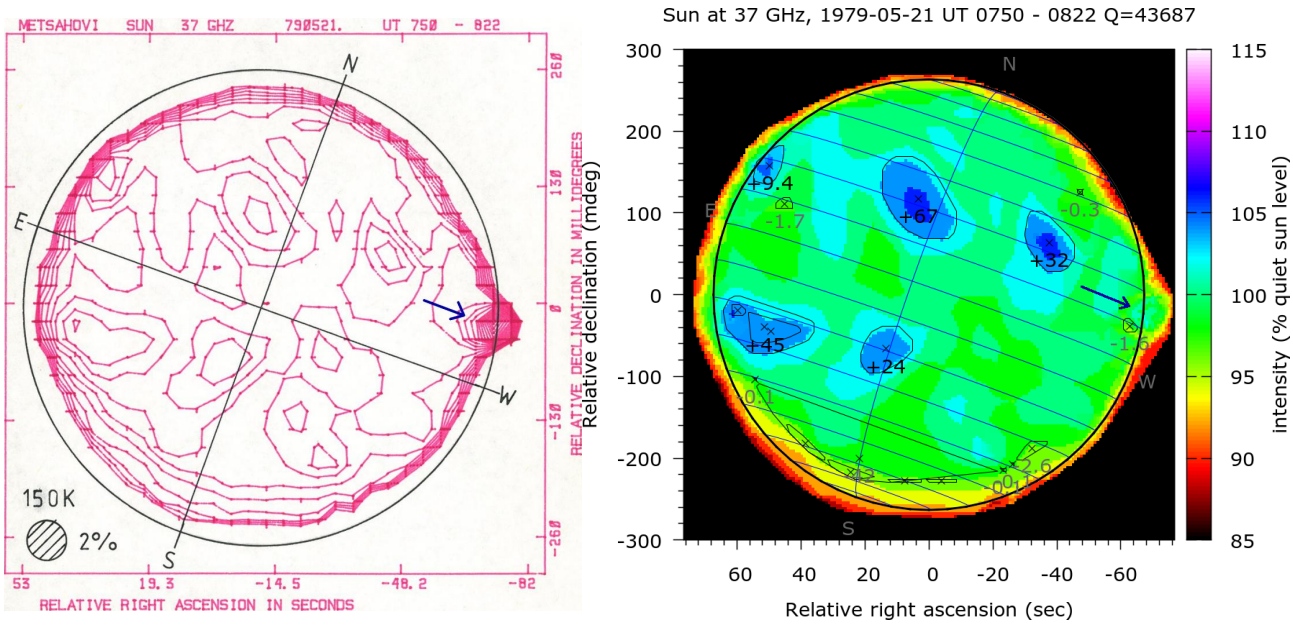

Figure 58: Brightening at the rim associated with a folding defect, marked by blue arrow.

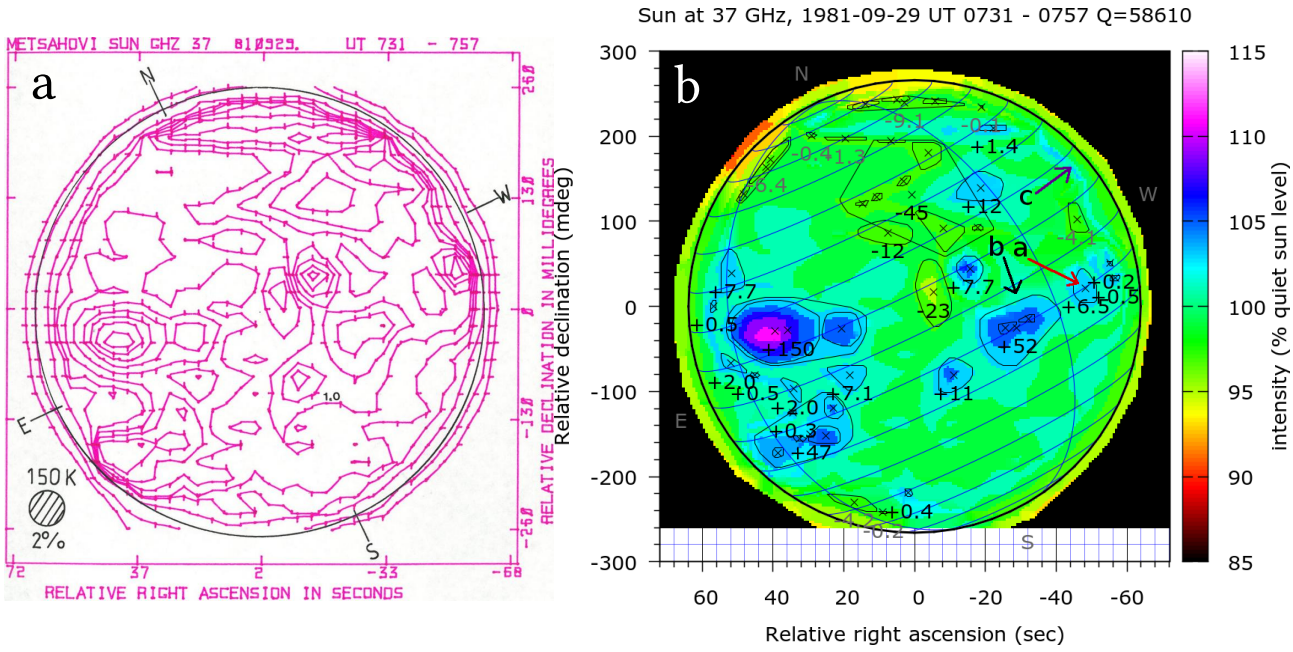

Figure 59: Mostly successful map with several detections of bright and dim regions. a) The code misses a gradient indicator, resulting in a a folding defect and a local maximum although it should be a minimum. b) Broken contour results in diffuse intensity field where there should be sharp transions and otherwise constant plateau areas. c) Wrong contour sign results in an elongated dim artefact and some intensity minima.

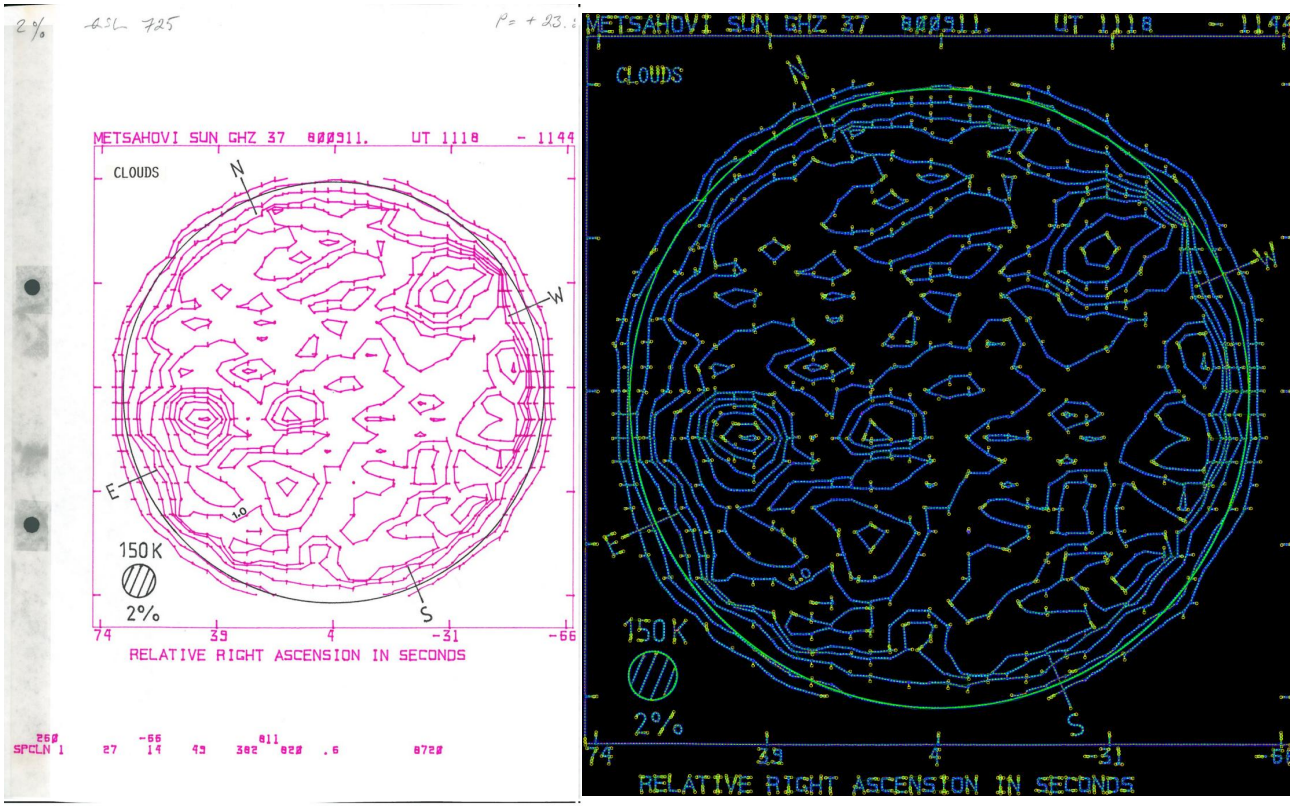

Figure 60: The labels for the y axis are not visible. Since the box is in an unexpected location in the document, preset cropping eliminated one boundary, and thus the layout could not be detected. This map will be fully recovered with correct cropping settings. The scale for declination will then be automatically calculated from the compass rose.

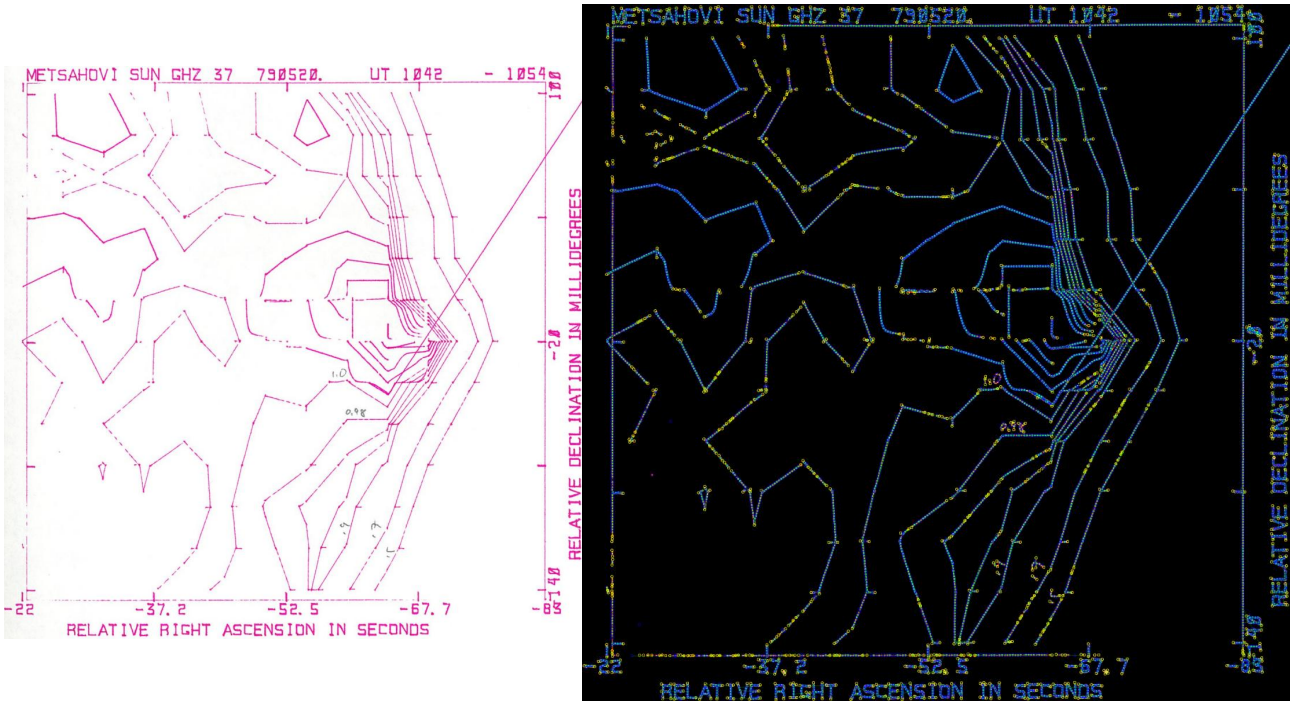

Figure 61: Weak ink and no layout detected.

4 Conclusion

Most of the maps are now successfully digitised, and the work continues in order to further improve the reliability. Metsähovi happily accepted preliminary results on March 8, 2018, and the whole data set was delivered during the next day. Even though many of the defects mentioned here could be fixed with minor effort, the scope of this work does not include further improvements after the delivery. Map quality had been improving continuously since December 2017, and this gradual process had to terminated at some arbitrary instant. This document already contains various suggestions on how to improve output quality. The mentioned defects and improvements will be considered as future work.

We have demonstrated various aspects of image processing, text recognition and optimisation. The scanned contour plots were filtered in order to reduce noise, after which a suitable histogram fraction was interpreted as pen paths. The layout of the map was interpreted, and any valuable information was stored as metadata. Contour lines were converted into scalar intensity field using a Poisson solver, and any broken contours were treated as perturbation to intact contours. At the region where this perturbation is weak, we substituted adjacent grid points to have mutually equal or offsetted values based on the local contour environment. This allowed us to iteratively eliminate the perturbation by reducing the amount of degrees of freedom in the system, while regularly solving the Poisson's equation again for our now modified system.

The obtained intensity maps were analysed for quality and statistics. The analysis produced a hiararchical structure of bright and dim intensity regions, which were also stored as metadata. Some preliminary analysis of the bright and dim regions was conducted. Our feature for detecting bright and dim region also serves the modern maps daily produced in Metsähovi.

This work also demonstrates various aspects of filtering, line detection, and optical character recognition. The cornerstone of many algorithms is the least squares method, which tries to find the best model for the system at hand by minimising a corresponding error function. When the local phase space may be approximated with a quadratic function, the quadratic method described here is suitable.

The code for processing the solar maps is sensitive to the chosen set of input parameters, which can be given at the command line. In order to find the best configuration, we must search the complicated and semi-discrete parameter space, for which an evolutionary optimisation scheme was implemented.

4.1 Ideas for further development

Many potential improvements suggested earlier are small modifications to the current code. Some wider concepts that require more development will improve

reliability and reduce parameter space. Less effort is then needed for finding suitable parameter combinations, and with improved approach, the code will be less sensitive to small modifications of parameters.

4.1.1 Line detection

With present software, once a point is added on a line it can no longer be removed. It is possible, at some occasions, that a line has grown to include points from two conflicting extremes. This happens when the input pattern has at least two slightly deviating branches with a common sraight root, and the algorithm is not sure which option to take. It will pick some of the closest points on both branches, until the average or maximum deviation from the mean line grows too large. The line fitting code will only include points when the resulting set will not produce a significant standard deviation from linear model. There is also a fixed maximum error that no point is is allowed to exceed. Thus afterwards, the code will be unable to adjust into additional points, even if one of the branches would clearly produce a straight line. It is then impossible to extend the line any further.

A solution could involve a conservative approach for removing points which deviate too much from the baseline. However, if the algorithm is too eager to drop such points, we might miss some meaningful lines from the pattern. Eternal loops will also be a problem, as the code may end up adding and removing the same points over again. The same phenomenon has been observed with path merging in 2.4. For this reason, it is important to have counters attached to each point, so that it can be added and removed from a line for only a limited number of times. When trying for every possible line combination, the computational load may sometimes grow unacceptable. This can be mitigated by indexing potential line candidates and trying only relevant combinations, thus reducing computational load.

4.1.2 Better adaptivity

At present, a single threshold level is used for the whole bitmap. This is a problem, since the map contains areas of different contour density. It might be useful to apply an adaptive contrast into the maps. In this approach, demonstrated with and example in (Fig. 62), the image is filtered in order to have uniform contrast level throught the image. After slight gaussian blur, a pre-defined threshold level is suitable for various exposure and lighting conditions. Since the line thickness may vary, each reference point will occupy as large area as possible. We hope this will increase reliability and reduce the parameter space compared to present implementation.

4.1.3 Character recognition

At present, the character recognition accepts either lines or pages of text. The pages are first split into multiple lines, and the lines are chopped into words, pairs

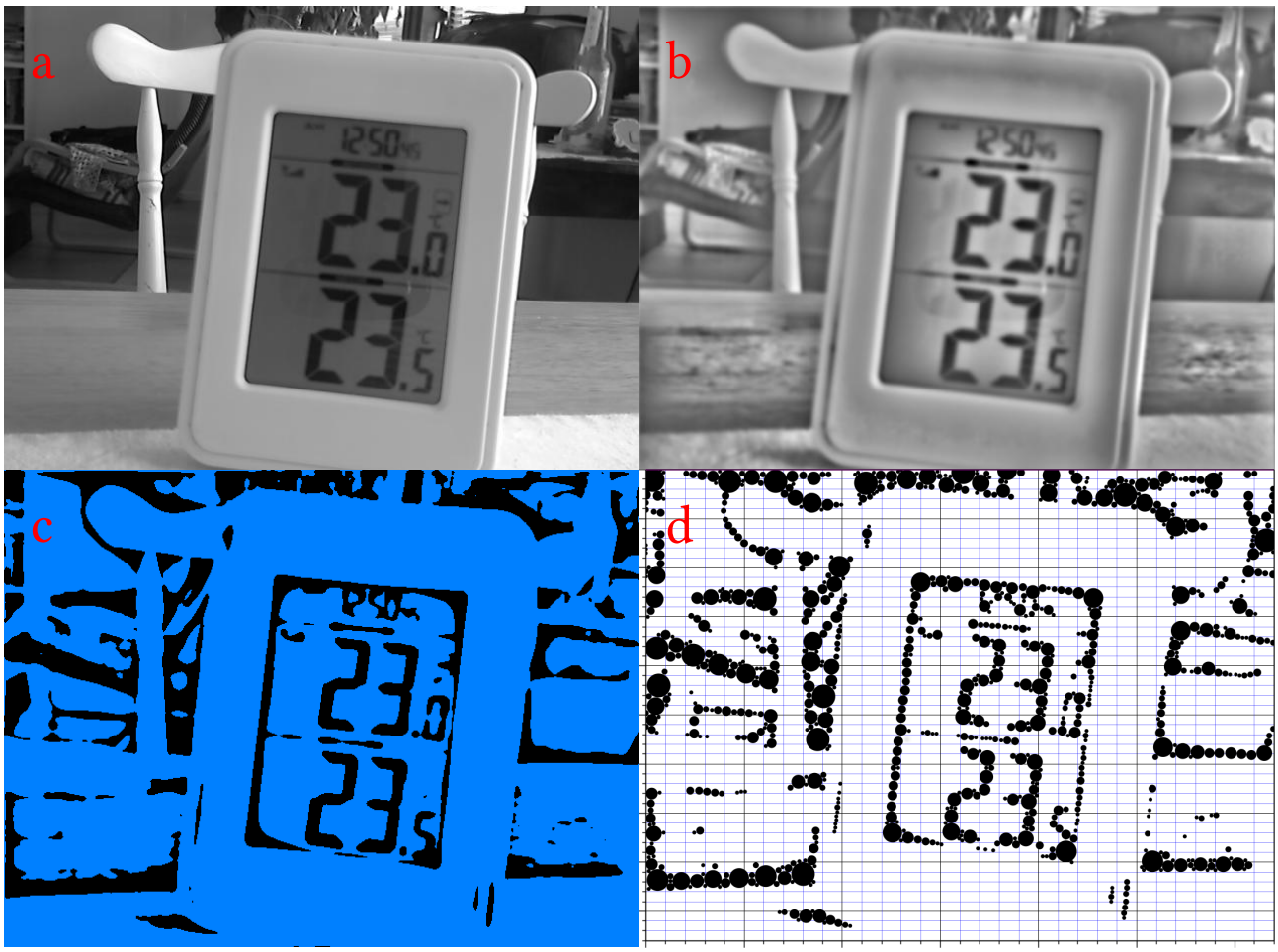

Figure 62: Adaptive contrast for line detection. a) Original image. b) After filtering and adaptive contrast. c) After applying a brightness threshold. d) After marking reference points with adaptive linewidth.

and individual characters using a set of parameters. This parameter set is tuned based on a collection of example inputs. Suitable values are critical for a successful result, and the matching is only sensitive to one particular typeface. This simple approach is unreliable, but could be improved by recognizing characters at arbitrary positions. The words and lines would then be constructed from the relative positions of characters. The matching algorithm presented in 2.6 could be augmented with a shallow neural network which would allow matching all the characters at once. That would offer more efficiency and less dependency on correct typeface selection.

4.1.4 Regularity in contours

Looking at Fig. 63, it becomes obvious that there is an underlying rectangular matrix of points, which is then interpolated in order to produce the contours seen in our historical contour plots. Contours have vertices at regular intervals both in x and y directions. An efficient method for analyzing these maps would then be first to search for regularity in the vertices, and detect the periodicity in a similar manner as was done for the gradient direction indicators. Then it would construct a grid of squares and fit line segments into the path data. Each line segment is treated only within one square, and contours are joined by connecting with compatible line segments in adjacent squares.

This approach however, does not necessarily apply to contour maps that are drawn manually or with a different interpolation algorithm. In its simplest form it only accepts line segments. It does not allow a contour segment to enter and leave a square with the same edge. Therefore, the research areas allowed by this approach would be limited. The idea of splitting the map into squares would still be useful.

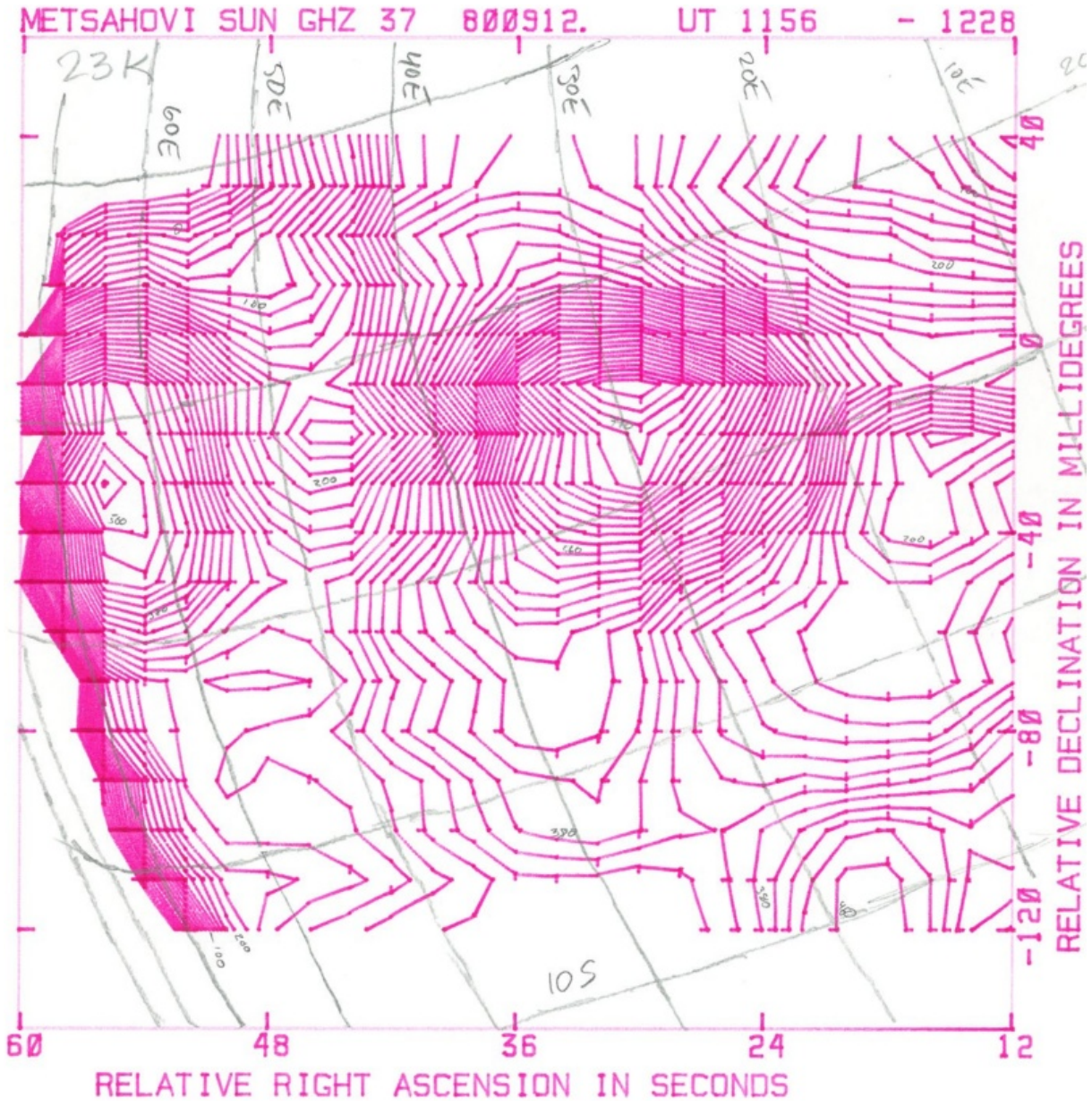

Figure 63: *Contours are based on a rectangular matrix of intensity values with linear interpolation.*

References

[1] S. Urpo (ed). URSI Commission J, Radio Astronomy in Finland 1987-1989.

[2] Metsähovi Radio Observatory web pages. `http://metsahovi.aalto.fi/en/research/sun/` (accessed June 4, 2018)

[3] MRO Solar Gallery. `http://www.metsahovi.fi/solar-gallery` (accessed June 4, 2018)

[4] M. Tornikoski. Ohjelma Auringon aktiivisten alueiden löytämiseksi. Tietämystekniikan erikoistyö, Teknillinen Korkeakoulu (1989).

[5] S. Urpo, S. Pohjolainen, H. Teräsranta. Solar Microwave Radiation Maps Measured at Metsähovi Radio Research Station in 1978-1979. Helsinki University of Technology, Metsähovi Radio Research Station A (1987).

[6] S. Urpo, S. Pohjolainen, H. Teräsranta. Solar Microwave Radiation Maps Measured at Metsähovi Radio Research Station in 1980-1981. Helsinki University of Technology, Metsähovi Radio Research Station A (1988).

[7] S. Urpo, S. Pohjolainen, H. Teräsranta. Solar Microwave Radiation Maps Measured at Metsähovi Radio Research Station in 1982-1983. Helsinki University of Technology, Metsähovi Radio Research Station A (1988).

[8] S. Urpo, S. Pohjolainen, H. Teräsranta. Solar Microwave Radiation Maps Measured at Metsähovi Radio Research Station in 1984-1986. Helsinki University of Technology, Metsähovi Radio Research Station A (1989).

[9] S. Urpo, S. Pohjolainen, H. Teräsranta, K. Karlamaa. Solar Microwave Radiation Maps Measured at Metsähovi Radio Research Station in 1987-1988. Helsinki University of Technology, Metsähovi Radio Research Station A (1991).

[10] S. Kivistö. Solar map digitisation code. Metsähovi Radio Observatory, Aalto University, Finland (2018). http://www.metsahovi.fi/~skkivist/

[11] The GNU Compiler Collection. Online documentation. https://gcc.gnu.org/onlinedocs/ (accessed June 6, 2018)

[12] The GNU C library. https://www.gnu.org/software/libc (accessed June 6, 2018)

[13] The GNU C Library. https://www.gnu.org/software/libc/manual/html_node/Mathematics.html#Mathematics (accessed June 6, 2018)

[14] The GCC low-level runtime library. https://gcc.gnu.org/onlinedocs/gccint/Libgcc.html (accessed June 6, 2018)

[15] The GNU C++ library. https://gcc.gnu.org/onlinedocs/libstdc++/ (accessed June 6, 2018)

[16] Linux Programmers Manual. Overview of the virtual ELF dynamic shared object. http://man7.org/linux/man-pages/man7/vdso.7.html (accessed June 6, 2018)

[17] Bash Reference Manual. https://www.gnu.org/software/bash/manual/bashref.html (accessed June 6, 2018)

[18] Netpbm - graphics tools and converters. https://packages.debian.org/source/stable/netpbm-free (accessed June 6, 2018)

[19] gnuplot homepage. http://gnuplot.info (accessed June 6, 2018)

[20] R. Howard, V. Bumba, S. F. Smith. Atlas of Solar Magnetic Fields. Mount Wilson and Palomar Observatories, California Institute of Technology (1967).

[21] ReSoLVE Center of Excellence Web Pages. http://www.spaceclimate.fi/resolve/about.html (accessed 4 June, 2018)

[22] M. Stix. The Sun, An Introduction (2nd edition, 2004).

[23] Kelvinsong / Wikimedia Commons / CC BY-SA 3.0 https://upload.wikimedia.org/wikipedia/commons/d/d4/Sun_poster.svg (accessed March 15, 2018)

[24] J. Kallunki. Studies of Solar Activity with Emphasis on Quasi-periodic Oscillations. Doctoral dissertation. Annales Universitatis Turkuensis. SER. A I TOM. 467. (2013).

[25] S. Elliot. The Physics and Chemistry of Solids. Wiley (1998).

[26] Lipson. Optical Physics (4th ed.), Cambridge University Press (2011).

[27] J. Kallunki, M. Tornikoski, J. Tammi, E. Kinnunen, K. Korhonen, S. Kesäläinen, O. Arkko. Forty Years of Solar Radio Observations at Metsähovi Radio Observatory. Astronomische Nachrichten (2018). https://doi.org/10.1002/asna.201813464 (accessed June 5, 2018)

[28] E. H. Avrett, R. Loeser. Models of the Solar Chromosphere and Transition Region from SUMER and HRTS Observations: Formation of the Extreme Ultraviolet Spectrum of Hydrogen, Carbon, and Oxygen. The Astrophysical Journal Supplement Series, 175:229Y276, 2008 March.

[29] SOHO Science Archive Interoperability Subsystem. http://ssa.esac.esa.int/ssa/aio/html/datarequests.shtml (accessed June 14, 2018)

[30] The Precision Solar Photometric Telescope (PSPT). http://lasp.colorado.edu/pspt_access/#data (accessed June 14, 2018)

[31] Robert A. Rohde / Wikimedia Commons / CC-BY-SA-3.0-migrated https://commons.wikimedia.org/wiki/File:Sunspot_Numbers.png (accessed March 15, 2018)

[32] David Hathaway / NASA / ARC. https://solarscience.msfc.nasa.gov/images/bfly.gif (accessed March 15, 2018)

[33] A. A. Norton, E. H. Jones, Y. Lui, K. Hayashi, J. T. Hoeksema, J. Schou. How Much More Can Sunspots Tell Us About the Solar Dynamo? IAU (2012). https://doi.org/10.1017/S1743921313002172 (accessed June 5, 2018)

[34] A. R. Choudhuri. Flux-transport and Mean-field Dynamo Theories of Solar Cycles. IAU (2012). https://doi.org/10.1017/S1743921313002184 (accessed June 5, 2018)

[35] A. Brandenburg. The Case for a Distributed Solar Dynamo Shaped by Near-surface Shear. The Astrophysical Journal, 625:539 âĂŞ547, 2005 May 20. https://doi.org/10.1086/429584 (accessed June 18, 2018)

[36] Metsähovi Radio Observatory web pages. http://metsahovi.aalto.fi/en/research/instruments/ (accessed March 15, 2018)

[37] S. Urpo, S. Pohjolainen, H. Teräsranta. Solar Radio Flares 1989-1991. Helsinki University of Technology, Metsähovi Radio Research Station, Series A, Report 11, 1992.

[38] HP Memory Project website. http://hpmemoryproject.org/wb_pages/wall_b_page_13.htm (accessed March 29, 2018)

[39] J. Kallunki, M. Tornikoski. Eruptive Solar Prominence at 37 GHz. Solar Physics (2017) 292:84. https://doi.org/10.1007/s1120701711107 (accessed June 10, 2018)

[40] M. Tornikoski. Ohjelmisto Auringon aktiivisuuden tutkimiseksi mikroaaltoalueella. Diplomityö. Teknillinen korkeakoulu (1990).

[41] Introducing JSON. http://www.json.org/ (accessed June 18, 2018)

[42] S. and S. Bazaraa. Nonlinear Programming, Theory and Algorithms (1993).

[43] D. Jurafsky, J. H. Martin. Speech and Language processing (2017).

[44] H. Modi, M. C. Parikh. A Review on Optical Character Recognition Techniques. International Journal of Computer Applications (0975 âĂŞ 8887) Volume 160 âĂŞ No 6, February 2017.

[45] F. Mohammad, J. Anarase, M. Shingote, P. Ghanwat. Optical Character Recognition Implementation Using Pattern Matching. International Journal of Computer Science and Information Technologies, Vol. 5 (2), 2014, 2088-2090.

[46] N. Venkata Rao, Dr. A.S.C.S.Sastry, A.S.N.Chakravarthy, Kalyanchakravarthi P. Optical Character Recognition Technique Algorithms. Journal of Theoretical and Applied Information Technology, Vol.83, No.2 (2016).

[47] A. Chudgor, V. Sawant. Implementation of Handwritten Character Recognition using Template Matching. International Journal of Current Engineering and Technology (2016).

[48] A. Pandey, S. Sawant, E. M. Schwartz. Handwritten Character Recognition using Template Matching. Florida Conference on Recent Advances in Robotics, FCRAR 2010 - Jacksonville, Florida, May 20-21, 2010.

[49] D. Nadeem, S. Rizvi. Character Recognition Using Template Matching. Project report. Department of Computer Science, Jamia Millia Islamia, New Delhi-25.

[50] M. Ryan, N. Hanafiah. An Examination of Character Recognition on ID card using Template Matching Approach. International Conference on Computer Science and Computational Intelligence (ICCSCI 2015).

[51] Rama C. Hoetzlein, Graphics Devtech, NVIDIA. Fast Fixed-radius Nearest Neighbors: Interactive Million-particle Fluids. GPU Techology Conference. http://on-demand.gputechconf.com/gtc/2014/presentations/S4117-fast-fixed-radius-nearest-neighbor-gpu.pdf (accessed June 10, 2018)

[52] The GNU OCR. Software package. https://www.gnu.org/software/ocrad/ocrad.html (accessed June 8, 2018)

[53] A. K. Mitra. Finite Difference Method for the Solution of Laplace Equation. Department of Aerospace Engineering, Iowa State University. http://www.public.iastate.edu/~akmitra/aero361/design_web/Laplace.pdf (accessed June 19, 2018)

[54] D. W. Williams. Probability With Martingales. Cambridge University Press (1991).

[55] D. R. Williams. Sun Fact Sheet. NASA Goddard Space Flight Center (2018). https://nssdc.gsfc.nasa.gov/planetary/factsheet/sunfact.html (accessed June 14, 2018)

[56] Solar-Terrestrial Coordinate Systems. The Wilcox Solar Observatory. http://wso.stanford.edu/words/Coordinates.html (accessed June 14, 2018)

[57] J. Kallunki, M. Uunila. Sizes of solar active regions at 8 mm. Astronomische Nachrichten (2016) https://doi.org/10.1002/asna.201613196 (accessed June 10, 2018)

[58] H. Karttunen, K. J. Donner, P. Kröger, H. Oja, M. Poutanen. Tähtitieteen perusteet. (URSA 2000)

A Black body radiation

Normal matter exhibits four fundamental interactions, which are gravity, the weak interaction, electromagnetism, and the strong interaction. There is an accurate theory for explaining electromagnetism as well as strong and weak interaction at the quantum level. Electromagnetic interaction is mediated by photons, which are massless particles and thus have a dispersion relation

$$\omega = c * k \qquad \text{where} \quad k = |\boldsymbol{k}| = \frac{2\pi}{\lambda} \qquad c = 2.997\,924\,58 \times 10^8\,\mathrm{m/s}. \tag{172}$$

In open space, there are a continuum of states for photons, while in a bounded environment the available quantum levels are discrete. In a cubic box of side length L, the available states are:

$$\psi(x,y,z) = C \sin(k_x x) \sin(k_y y) \sin(k_z z) \tag{173}$$

$$k_i L = \pi n_i \qquad \text{for} \quad i = x,y,z \qquad n_x, n_y, n_z = 1,2,..., \tag{174}$$

where each photon has energy $E = \hbar\omega$, so that there are $\Omega(E_{\max})$ states available with energy at most $E_{\max}$:

$$n_{\max} = \frac{L E_{\max}}{\pi c \hbar} \qquad n_x^2 + n_y^2 + n_z^2 \leq n_{\max}^2. \tag{175}$$

Given that the L is sufficiently large, we can include states within an octet of a sphere, and remember that there are two independent polarization directions:

$$\Omega(E_{\max}) = 2 * \frac{4\pi n_{\max}^3}{3 * 8} = \frac{L^3 E_{\max}^3}{3\pi^2 c^3 \hbar^3}. \tag{176}$$

The density of states becomes:

$$d(E) = \frac{\mathrm{d}\Omega(E)}{\mathrm{d}E} = \frac{L^3}{\pi^2 c^3 \hbar^3} E^2. \tag{177}$$

Photons are bosons, so that there may be any integer number particles occupying each state. As the box is connected to its environment which has temperature T, the propability of having n photon occupying a particular state with energy E is:

$$p(E,n) = \frac{e^{-\frac{nE}{k_\mathrm{B}T}}}{\sum_{j=0}^{\infty} e^{-\frac{nE}{k_\mathrm{B}T}}} = e^{-\frac{nE}{k_\mathrm{B}T}} \left(1 - e^{-E/k_\mathrm{B}T}\right), \tag{178}$$

and the average number of photons on that state is:

$$< n > (E) = \left(1 - e^{-E/k_\mathrm{B}T}\right) \sum_{j=0}^{\infty} j * p(E,j). \tag{179}$$

We thus need to calculate series $S_x = \sum_{j=0}^{\infty} j x^j$, so that:

$$(1-x) S_x = \sum_{j=1}^{\infty} j x^j - \sum_{k=1}^{\infty} k x^{k+1} = \sum_{j=1}^{\infty} x^j = \frac{x}{1-x}, \tag{180}$$

and

$$< n > (E) = \frac{e^{-E/k_{\mathrm{B}}T}}{1 - e^{-E/k_{\mathrm{B}}T}} = \frac{1}{e^{E/k_{\mathrm{B}}T} - 1}. \tag{181}$$

Number of photons occupying a narrow interval of energy $[E, E + \mathrm{d}E]$ is then

$$n(E) = \frac{L^3 E^2}{\pi^2 c^3 \hbar^3} * \frac{1}{e^{E/k_{\mathrm{B}}T} - 1} \mathrm{d}E, \tag{182}$$

with their energy density (in volume L^3) related to frequency $\nu = \omega/2\pi$ as:

$$\epsilon = \frac{8\pi\nu^3 h}{c^3} * \frac{1}{e^{\frac{h\nu}{k_{\mathrm{B}}T}} - 1} \mathrm{d}\nu. \tag{183}$$

The radiation is propagating in all directions (4π steradians) with speed c, so that for one $L \times L$ face of the box and an incident angle θ, the round trip time for a photon is $\frac{2L}{c*|\cos\theta|}$. The total power incident on the face is:

$$\frac{L^3\epsilon}{4\pi} \int_{\theta=0}^{\pi} \int_{\varphi=0}^{2\pi} \sin\theta \left| \frac{2L}{c * \cos\theta} \right|^{-1} \mathrm{d}\varphi \, \mathrm{d}\theta = \frac{L^2 \epsilon c}{2} \int_{\theta=0}^{\pi/2} \sin\theta \cos\theta \, \mathrm{d}\theta \tag{184}$$

$$= \frac{L^2\epsilon c}{2} \left| \frac{-1}{4} \cos 2\theta \right|_{\theta=0}^{\pi/2} = \frac{L^2 \epsilon c}{4}, \tag{185}$$

thus the spectral power per surface area is:

$$I(\nu) = \frac{2\pi h \nu^3}{c^2} \frac{1}{e^{\frac{h\nu}{kT}} - 1} \mathrm{d}\nu. \tag{186}$$

When we integrate over the whole spectrum, se will obtain the Stefan-Boltzmann law:

$$\int_{\nu} 0 \, \mathrm{d}\nu \infty \frac{\nu^3}{e^{\alpha\nu} - 1} = \alpha^{-4} \int_{u} 0 \, \mathrm{d}u \infty \frac{u^3}{e^u - 1} = \left(\frac{kT}{h} \right)^4 \frac{\pi^4}{15}, \tag{187}$$

and the total intensity is

$$I_{\mathrm{total}} = \frac{2\pi^5 k^4}{15 c^2 h^3} T^4 = \sigma T^4. \tag{188}$$

The frequency with maximum intensity can be obtained as:

$$\frac{\mathrm{d}}{\mathrm{d}\nu} \frac{\nu^3}{e^{\alpha\nu} - 1} = 0 \quad \Rightarrow \quad 3\nu^2 \left(e^{\alpha\nu} - 1 \right) - \nu^3 \alpha e^{\alpha\nu} = 0, \tag{189}$$

so that:

$$e^{\alpha\nu} - 1 = \frac{\alpha\nu}{3} e^{\alpha\nu} \qquad \alpha\nu := 3 - 3e^{-\alpha\nu} \qquad \nu = 2.8214 * \frac{kT}{h}, \tag{190}$$

which is the Wien's displacement law.

When the frequency is very low, that is $\nu \ll \frac{kT}{h}$, we can approximate $e^{\alpha\nu} \approx 1+\alpha\nu$ and obtain the Rayleigh-Jeans law:

$$I(\nu) = \frac{2\pi k\nu^2}{c^2}T, \tag{191}$$

which conveniently allows us to use temperature scale for intensity and noise levels.

B List of command line parameters

The parameters can be grouped based on their typical usage. This features the current status of the parameter list, which will develop further. The parameter list is defined in the source file *cfparams.cpp* and the related headers. With each parameter, the default value is supplied.

B.1 Location of the observer

These describe the location of the observer on Earth. Currently it is based on spherical latitude and longitude, which do not take into account the flattening of the Earth. The model will also be improved to include the Earth orbiting the common centre of mass with the Moon. The Moon's orbit follows Cassini's laws and are complicated due to tidal effects from the Sun [58]. These improvements, however, will not affect at the level of the accuracy of the solar maps and will be needed only for formal consistency.

observer_lat	60.217797339	Spherical latitude of the observer (currently used Metsähovi radio observatory geographic latitude)
observer_lon	24.393084663	Spherical longitude of the observer (currently used Metsähovi radio observatory geographic longitude)

B.2 Miscellaneous options for output, debugging, testing and demonstrations

These are mainly hooks used when developing and documenting the digitisation process.

store_bitmap	0	Output the filtered image with recognized map features included
filtertest	0	Apply filters and quit
pathtest	0	Detect tips and/or sledge paths and quit
chaintest	0	Build paths and quit
mergetest	0	Perform line and arc merging and quit
do_fill	1	After all tips and sledge paths are tracked, try to fill in more points to connect remaining lines
do_symbol_templates	0	Use the map to extract symbol templates to be used later

template_grayscale	1	Templates are based on greyscale instead of threshold
debug_convergence	0	Produce a heatmap for each step of convergence when reducing degrees of freedom in the Poisson's equation
nice	1	Wait until CPU idle and nobody logged in

B.3 Miscellaneous output modifiers

These specify the desired output and give some general information about the input.

metadata_documentation	1	Include metadata comments in JSON file
do_heatmap	1	Produce an intensity map using gnuplot
do_filtered	0	Produce a filtered intensity map using gnuplot
do_carrington	1	Produce a Carrington map using gnuplot
do_gnuplot	0	Produce a gnuplot script which is able to render the .map file
tmplog	1	Store stderr logs to temporary file copy them directly to filesystem (either in the date directory or failures
quiet	0	Do not produce error output when processing files
write_outputs	1	Store files if successful (when optimizing we don't want to store any files)
cosine_corrected_img	1	Assume that the scanned maps are scaled such that the sun looks like a circle
cosine_corrected_map	1	Assume that the relative right ascension on the map is cosine-corrected
reliable_ticks	0	Limit convex hull based on tick coordinates only
colorbox	1	Include the colour box in gnuplot renderings (it is sometimes avoided in collages)

B.4 Filtering done in Fourier space

Filtering reduces noise and adapts to different brightness levels.

do_bandpass	1	Filtering with gaussian-shaped transfer function
do_RLC	0	Filtering which simulates an electrical circuit with a resistor, inductor and a capacitor
do_absgrad	0	Calculate absolute gradient of the filtered image, which is good for detecting edges
do_laplacian	0	Calculate laplacian of the filtered image, which is also suitable for detecting certain types of edges
filter_phase	0	Apply a phase shift in all filtered frequencies
lambda_min	7.2	Compares to how short wavelengths are allowed in the filtered image
lambda_max	64.7	Compares to how long wavelenghts are allowed in the filtered image
imagethreshold	1	This is for visualizing the filtered image

B.5 Creating reference points

The filtered greyscale image will be covered with reference points where the ink strength is above certain threshold. which occupy a circular area. They are assigned starting where the ink is strongest.

fraction	0.067	Histogram fraction available for white circles
cpoint_dist	8	Minimum distance between white circles

B.6 Tip detection

Tips need to be detected before the path detection can work reliably.

tip_crown_dist	7.9	Distance in study when detecting tips
tip_head_dirs	36	How many intermediate points to use in tip head detection
tip_crown	9	How many points, in two directions, should be free
tip_optimize_dist	0.3	How small steps to use when optimizing tip location after crude detection
tip_neck_dist	12.9	Distance to study when extending tip neck
tip_neck_angle	74.0	Total angle available for extending tip neck

tip_neck_dirs	22	Intermediate directions for calculating minima when extending tip neck
tip_neck_width	4	How many points, in two directions, to include when calculating tip neck minima
tip_max	6	How many times a tip neck can be extended

B.7 Path detection

Sledge algorithm tries to stay on contours lines.

line_init_dirs	36	Intermediate directions when detecting a pair of opposite directions for a local path
line_init_width	3	How many points, in two directions, to include when calculating minima for local path
sledge_dist	11.4	Distance to study ahead when advancing on a local path
sledge_angle	157.4	Total angle available for studying the local path when advancing
sledge_dirs	22	Intermediate points when calculating minima when advancing on a path
sledge_width	1	How many points, in two directions, to include when calculating a particular minima for a step
sledge_factor	12	How much to favour the original direction and go straight

B.8 Optimizing tip and sledge

A fine-tuning step is added after every tip detection or sledge advance.

optimize_angle	21.4	Optimize path tracking direction within this angle with respect to a crude direction
optimize_dirs	15	How many intermediate points to use in the direction optimization

B.9 Building chains and contours from reference points

Once the reference points are set using the sledge algorithm, neighbouring points are connected according to proximity and trying to favour straight paths, based on priority sorting. After chains are constructed, they are further joined into

contour lines once the layout is determined.

neighbour_dist	60	When mapping neighbouring white circles, consider up to this distance
conn_dist	16.3	Distance allowed when to white circles can be connected as a chain or contour, or proto-line
distance_factor	1.10	How much to favour short knees over straight knees, in chains and contours
penalty_factor	1.25	How much to penalize potential gradient indicators when building a chain and contour (factor)
penalty_offset	0	How much to penalize potential gradient indicators when building a chain and contour (fixed offset)
chain_angle_max	92.9	Maximum angle a chain can take
chain_points_min	4	Minimum points in a chain
contour_points_min	8	Minumum points in a contour before splitting
contour_hull_min	10	Minimum points in a contour when it can be included in the convex hull
chain_merge_tolerance	5	How many points (white circles) can be scrapped when joining two chains to build a contour
trouble_threshold	2	If there are at least this many additional connected neigbours for a white circle, it is considered as a troubled one
greedy_merging	1	Merge only when the resulting total length increases
restrict_indicators	1	Allow gradient indicators only when there is a T-junction (tip) detected on the bitmap level

B.10 Line recognition

These parameters determine how straight path segments are interpreted as lines. Shorter lines are merged into longer ones when they are compatible.

line_expand	25	Distance to consider for new points when extending a line

line_stdev	3	Maximum standard deviation allowed for a detected straight line
line_error	8	Maximum point error allowed for a straight line
line_error_new	3	Maximum initial error allowed when trying to include more points in a straight line
line_merge_long_threshold	400	If there are two line which are at least this long, they can be joined even if they are not within distance 'line_expand' apart
line_merge_long_expand	60	This is the distance considered when joining two long lines
line_points_min	4	Minumum number of points in a line

B.11 Arc and circle detection

After lines have reached maximum length by merging, the code tries to interpreted intersecting lines as arc segments. Merging is also applied to arcs.

arc_expand	35	Distance to consider when joining to arcs
arc_stdev	2	Maximum standard deviation allowed for an arc
arc_error	15	Maximum point error allowed for an arc
arc_error_new	3	Maximum initial error allowed when trying to include more points in an arc
circle_points_min	30	Minimum number of points allowed in a circle
arc_radius_min	40	Minimum radius of a circle
arc_radius_max	1000	Maximum radius of a circle
circle_coverage	0.8	Fraction of circle perimeted required to be covered with white circles

B.12 Map boundary box

These are tolerances related to map boundary box.

box_points_min	200	Box lines need to have at least this many points
box_angle_tol	3	Box corners must be 90 degrees, within tolerance this many degrees

box_length_min	500	Box lines need to be at least this long
box_length_tol	20	Box lines must have equal length, within tolerance specified here

B.13 Compass rose or cross

These are tolerances and other definitions related to the compass rose or cross. The required lines and the big circle originate from the merging stage.

compass_rose_min_radius	500	Compass rose, the big circle, must have radius at least this
compass_rose_min_cross	700	Crossed lines must be at least this long, even if there is no big circle
compass_rose_indicator_angle_tol	15	Direction indicators must be radially oriented, within tolerance of this many degrees
compass_rose_compound_angle_tol	3	Two long lines crossed, or four short lines, must be mutually perpendicular or antiparallel, within tolerance of this many degrees
compass_rose_radial_tol	50	Direction indicators much touch the big circle, within tolerance of this many pixels
compass_rose_min_length	50	Direction indicators must be at least this long
compass_rose_pos_tol	60	If there is a big circle, the crossed lines must intersect at the centre of the big circle, within tolerance specified here
compass_rose_intersect_tol	50	When N-S and W-E axes do not intthe dependencyersect within the box, the axes must end at the box boundary, within tolerance specified here

B.14 Compass rose direction indicator symbols N W S E

These are drawn with stencils and are aligned with the compass rose direction indicators.

nwse_sep	112	How far the centre of direction symbols N W S E typically is from the tip of a direction indicator
nwse_width	100	Width needed for a direction symbol N W S E
nwse_height	200	Height needed for a direction symnol N W S E, taking into account the possibility that part a symbols letter may be considered as part of a direction indicator line
nwse_tol	10	Tolerance when matching white circles to N W S E templates
nwse_scale_min	0.7	Minimum scale when matching N W S E symbols with the templates
nwse_scale_max	1.3	Maximum scale when matching N W S E symbols with the templates
nwse_max_error2	30	Maximum total error allowed when matching N W S E symbols with the templates

B.15 Antenna beam size size indicator

This may either contain tilted lines or the numerical indicator label, which represents the antenna beam diameter in arc minutes. There is more contour information near the antenna beam size indicator, and two different stencil sizes are used from drawing them.

lobe_min_radius	40	Antenna beam must have radius at least this
lobe_scanline_tol	20	Scanlines in the beam size indicator must touch the circle perimeter, within tolerance of this many pixels
lobe_angle_tol	5	Scanlines must be parallel, within tolerance of this many degrees
lobe_corner_dist	400	The beam size indicator circle must be close to map corner, with maximum distance of the beam size indicator centre specified here
lobe_min_lines	3	There must be at least this many scanlines in the beam size indicator circle, otherwise it is considered as text
lobe_default_at_37GHz	0.02	Default beam radius when frequency is 37 GHz

default_frequency_GHz	37	Default frequency in GHz
lobe_shrinkage	0.75	When fitting an empirical disk/ellipse, shrink by this amount relative to beam radius

B.16 Interpreting contour level separation

The contour level spacing labels are accompanied with the antenna beam size indicator.

lobetext_sep	10	Minimum distance between beam size text and indicator circle perimeter
lobetext_width_max	270	Maximum width of texts accociated with the beam size indicator
lobetext_height_min	60	Height of beam texts (one line)
lobetext_height_max	250	Height of beam texts (two lines)

B.17 Tick markers and labels in axes RA and dec

Tick markers are short line segments which point into the map area. Below them are mechanically drawn numbers which are either seconds of right ascension or millidegrees of declination.

tick_angle_tol	20	Ticks (which specify map coordinates) must be perpendicular to the box boundaries, within tolerance of this many degrees
tick_length_min	15	Mininum length for a tick
tick_length_max	40	Maximum length for a tick
tick_pos_tol	20	Positional tolerance for ticks and the relative scale they are referring to
tick_sep	7	Distance between the box boundary and the tick label below it
tick_label	75	Space reserved for the tick label texts, so that the description texts are on the next line
tick_width	250	Width of tick label texts
tick_height	60	Height of tick label texts and other texts drawn with similar typeface

B.18 Text drawn by the mechanical plotter

These parameters specify how the text line is split into individual words, sometimes pairs, and finally into individual characters.

tick_word_width_min	30	Minimum word length in pixels
tick_word_width_max	500	Maximum word length in pixels
tick_word_space	12	Space required between two words
tick_char_width_min	11	Minimum width of a character
tick_char_width_max	65	Maximum width of a pair of characters
tick_char_space	1	Space between a characters or pairs of characters
tick_elem_width_min	6	Minimum width of a character
tick_elem_width_max	36	Maximum width of a character

B.19 Character recognition

These parameters define how to match cropped bitmaps into template characters.

tick_radius	5	Radius use for pairing tick letters with their templates
tick_magnitude	45	Percent histogram fraction used for characher recognition
tick_tol	4	Tolerance used when pairing tick letters with their templates
tick_scale_amt	20	Number of different equally spaced scaling values used when comparing with templates
tick_scale_min	0.85	Minimum scale used when comparing with templates
tick_scale_max	1.15	Maximum scale used when comparing with templates
tick_score_max	5	Maximum mean square error allowed when recognizing a character

B.20 Adaptation to different plotter typeface

Sometimes the mechanical plotter has been configured to draw slightly different text. This typeface shares some of the characters with the other typeface, while some characters are larger. The same character templates are used with this

typeface, while different parameters are required for locating the characters.

louse_word_width_min	40	Minimum word length in pixels
louse_word_width_max	500	Maximum word length in pixels
louse_word_space	12	Space required between two words
louse_char_width_min	11	Minimum width of a character
louse_char_width_max	75	Maximum width of a pair of characters
louse_char_space	2	Space between a characters or pair of characters
louse_elem_width_min	5	Minimum width of a character
louse_elem_width_max	40	Maximum width of a character

B.21 Stencil typefaces

Markings related to the antenna beam size indicator and contour level spacing are drawn stencil-assisted by hand. Sometimes also the mechanically drawn text has been later corrected with stencil markings. There are two different stencil sizes used for this purpose, but they share the line properties.

stencil_line_height_min	40	Minimum line height (pixels) for stencil-drawn lines
stencil_line_height_max	80	Maximum line height (pixels) for stencil-drawn lines
stencil_line_space	10	Space required between two stencil-drawn lines
stencil1_word_width_min	60	Sometimes the tick labels are overwritten using stencil-drawn numbers, so that a different parameters set is needed for them
stencil1_char_width_min	15	Minimum width of a character
stencil1_char_width_max	40	Maximum width of a pair of characters
stencil1_char_space	1	Space between a characters or pair of characters
stencil1_elem_width_min	100	Minimum width of a character
stencil1_elem_width_max	100	Maximum width of a character

stencil1_scale_amt	20	Number of different equally spaced scaling values used when comparing with templates
stencil1_scale_min	0.60	Minimum scale used when comparing with templates
stencil1_scale_max	1.50	Maximum scale used when comparing with templates
stencil1_score_max	20	Maximum mean square error allowed when recognizing a character
stencil2_word_width_min	60	Recognize contour separation in temperature (e.g. 150 K) and in percents relative to the Quiet Sun Level (e.g. 2%)
stencil2_char_width_min	15	Minimum width of a character
stencil2_char_width_max	90	Maximum width of a pair of characters
stencil2_char_space	1	Space between a characters or pair of characters
stencil2_elem_width_min	100	Minimum width of a character
stencil2_elem_width_max	100	Maximum width of a character
stencil2_scale_amt	20	Number of different equally spaced scaling values used when comparing with templates
stencil2_scale_min	0.60	Minimum scale used when comparing with templates
stencil2_scale_max	1.05	Maximum scale used when comparing with templates
stencil2_score_max	20	Maximum mean square error allowed when recognizing a character

B.22 Gradient direction indicators

Gradient direction indicators determine the sign of a contour, and they point outwards when the intensity within the contour curve is higher than its surroundings.

gradient_indicator_linewidth	5.80	Horizontal tolerance for gradient direction indicators

gradient_indicator_length_min	5.18	Minimum length of gradient direction indicators
gradient_indicator_length_max	22.79	Maximum length of gradient direction indicators
gradient_indicator_safe_distance	24.49	In order to detect a gradient indicator with sufficient precicion, there must be open space up to this distance
gradient_indicator_safe_angle	59.70	The contour must be perpendicular to the indicator, or it has to make at least an angle specified here (degrees)
gradient_indicator_contour_dmin	9.64	When distinguishing contour from the gradient direction indicator, use this minimum distance
gradient_indicator_contour_dmax	19.40	When distinguishing contour from the gradient direction indicator, use this maximum distance
gradient_indicator_contour_angle	101.84	The contour must not make a too sharp angle right at the indicator, so this is the minimum angle in degrees
gradient_indicator_grid_linewidth	9.58	When the gradient indicators are required to follow periodic order, this is the maximum deviation allowed from the periodicity

B.23 Convex hull and empirical solar disk

The code tries to recognize the visible solar disk from the contours. This is done by first collecting the convex hull of reasonable contours, and then by detecting arc segments. Partially visible disk does not have a full circle.

hull_length_min	400	Arc in a convex hull must be at least of this length of there is no cutoff
hull_length_max	700	If the convex hull has a long straight line, is is probably due to limited data

hull_stdev	10	Maximum standard deviation allowed when there is such artefact arising from limited data
hull_error	20	Maximum point error when there is an artefact of limited data in the convex hull
disk_pos_tol	0.05	If the centre of solar disk is out of place by this amount, try another way to resolve coordinates
disk_rad_tol_relative	0.2	Relative tolerance in disk radius
empirical_circle_dev	0.010	When map boundary intersects with the circle fitted to convex hull, within this tolerance, the map is cropped (visual degrees)
disk_cutoff	0.06	Contours that are by this relative amount away from the theoretical disk, can be scrapped if the coordinates are certain (visual degrees)
bright_feature_at_perimeter	0.025	Contours that are by this relative amount away from the empirical disk can be scrapped (this is slightly less than what bright features may have at the perimeter, since the bright features themselves cause the empirical disk to have larger average radius, degrees)
dark_feature_at_perimeter	0.035	Areas close to the perimeter can not be used for calculating Quiet Sun Level and are never considered as dark features
default_arcsec_per_pixel	1.13	Arcseconds per pixel when there are no other means for determining it
convex_hull_iterations	10	Maximum number of iterations when determining the convex hull and eliminating apparent defects

B.24 Contour interpretation

Contour map is converted into scalar intensity map using Poisson solver and dimensionality reduction.

contour_grid_n	200	How many matrix columns and dows
shrink_dim	8000	Initially shrink degrees of freedom up to this
contour_compress_tol	0.10	Merge areas which agree in contour value up to this tolerance
shrink_factor	0.5	Try to reduce degrees of freedom into this relative amount during each step of convergence, increasing this closer to 1 increases required steps, but results in better quality
perimeter_width	70	Contours close to the convex hull are oriented along the perimeter are required to have positive sign
perimeter_angle_max	45	This is how well, in degrees, the contour must be parallel to the nearby convex hull in order to have fixed sign
contour_lowpass_lambda_half	0.02	Gaussian low pass filter for the intensity map to elimnate sharp contour edges, this is the wavelength with intensity halved
contour_lowpass_gain	1	Gain at low wavelenghts

B.25 Finding bright and dim regions

Bright and dim regions as well as the Quiet Sun Level are identified based on statistical analysis or external input. Fixed thresholds can be given.

subregion_cells_min	5	Subregions must contain at least this many cells
QSL_contour_min	0	If nonzero, this is contour levels above black sky for QSL minimum threshold

QSL_contour_mid	0	If nonzero, this is contour levels above black sky for QSL middle value
QSL_contour_max	0	If nonzero, this is contour levels above black sky for QSL maximum threshold
first_contour_QSL	0.89	Threshold value for the first contour, as a fraction of Quiet Sun Level, used only when it is not obtained from text recognition
QSL_default_K	7500	Quiet Sun Level in Kelvins, when it is not obtained from text recognition
QSL_forced_between_contours	1	Statistical Quiet Sun Level is assumed midway between two adjacent contour levels

B.26 Interpolating the modern solar map format (*.sunmap)

The code is also able to detect bright and dim regions from antenna data. The data has to be converted into rectangular matrix, so various methods of interpolation are supplied here.

interpolate_freedom	0.20	Relevant for analytic basis: maximum number of degrees of freedom relative to the amount of measurements
interpolate_cell_factor	15	Relevant non-analytic basis: when using finite elements, how many measurement points, on average, to allocate for each element
interpolate_tol_QLUP	10^{-15}	Tolerance used in QLUP decomposition of an underconditioned linear system
interpolate_type	-1	Type of basis functions used (-1: non-analytic piecewise polynomials 0: sin and cos, 1: polynomials of x and y)
interpolate_deg_max	7	Relevant for non-analytic basis: degree of polynomials used in finite elements
interpolate_diff_max	2	Relevant for non-analytic basis: degree of highest continuous differential in boundaries of finite elements

interpolate_wigner	0	Use Wigner-Seitz cells as finite elements (non-analytic basis)
interpolate_grid	0	Use rectangular grid cells as finite elements (non-analytic basis)

B.27 Generating the output

The code is able to produce map files which resemble antenna data, even through the actual sweep information is lost. This allows simple scripts to be able to process all maps. By default, simple rectangular output is preferred.

number_of_sweeps	100	Nonzero value here will give an output with a fixed number of sweeps
output_sample_aspect_ratio	1	Sample width spacing / sample height spacing
sweep_margin	0	If the data is not cropped in that direction, simulate antenna changing direction between sweeps
linear_QSL	10000	Linear power observed at Quiet Sun Level
linear_offset	0	Linear power observed at black sky
logarithmic_QSL	10000	Logarithmic power observed at 100% of the Quiet Sun Level
logarithmic_101	10100	Logarithmic power observed at 101% of the Quiet Sun Level
logarithmic_noise	0	Logarithmic power observed at black sky
carrington_offset	0	Left-most longitude in Carrington plots

C Detecting periodic Dirac comb pattern

When a signal window $[0, 2\pi]$ contains a sinusoidal signal $\cos(n\omega t + \omega_0)$ for some integer n, we observe a single sharp peak in its Fourier transform. An ill-tuned peak has non-harmonic, but fractional, multiple frequency compared to the base frequency $n = 1$. The peak covers many channels instead of a just one, and the amplitudes of the channels follow sinc function:

$$\begin{aligned}
\hat{f}_k &= \int_{t=0}^{2\pi} e^{ikt} \cos(\omega_0 t - \alpha)\, \mathrm{d}t && (192)\\
&= \frac{e^{-i\alpha}}{2} \int_{t=0}^{2\pi} e^{it(k+\omega_0)}\, \mathrm{d}t + \frac{e^{i\alpha}}{2} \int_{t=0}^{2\pi} e^{it(k-\omega_0)}\, \mathrm{d}t && (193)\\
&= \frac{e^{-i\alpha}}{2i(k+\omega_0)} \Big|_{t=0}^{2\pi} e^{it(k+\omega_0)} + \frac{e^{i\alpha}}{2i(k-\omega_0)} \Big|_{t=0}^{2\pi} e^{it(k-\omega_0)} && (194)\\
&= e^{-i\alpha} \frac{e^{2\pi i(k+\omega_0)} - 1}{2i(k+\omega_0)} + e^{i\alpha} \frac{e^{2\pi i(k-\omega_0)} - 1}{2i(k-\omega_0)}. && (195)
\end{aligned}$$

This further becomes:

$$\begin{aligned}
&= e^{-i\alpha} e^{i\pi(k+\omega_0)} \frac{e^{i\pi(k+\omega_0)} - e^{-i\pi(k+\omega_0)}}{2i(k+\omega_0)} + e^{i\alpha} e^{i\pi(k-\omega_0)} \frac{e^{i\pi(k-\omega_0)} - e^{-i\pi(k-\omega_0)}}{2i(k-\omega_0)} && (196)\\
&= e^{i(\pi(k+\omega_0)-\alpha)} \frac{\sin(\pi(k+\omega_0))}{k+\omega_0} + e^{i(\pi(k-\omega_0)+\alpha)} \frac{\sin(\pi(k-\omega_0))}{k-\omega_0} && (197)\\
&= \pi e^{i(\pi(k+\omega_0)-\alpha)} \operatorname{sinc}(\pi(k+\omega_0)) + \pi e^{i(\pi(k-\omega_0)+\alpha)} \operatorname{sinc}(\pi(k-\omega_0)). && (198)
\end{aligned}$$

Since k is an integer close to ω_0, we can approximate:

$$\hat{f}_k \approx \pi e^{i(\pi(k-\omega_0)+\alpha)} \operatorname{sinc}(\pi(k-\omega_0)) \quad (199)$$

Let $j, k \in \mathbb{Z}_+$ so that $\omega_0 \in (k-1, k,)$. Then:

$$\hat{f}_j = \pi e^{i(\pi(k-\omega_0)+\alpha)} \frac{|\sin(\pi(j-\omega_0))|}{\pi(j-\omega_0)} \qquad \hat{f}_j = \frac{k-\omega}{j-\omega} \hat{f}_k, \quad (200)$$

and

$$j\hat{f}_j - \omega\hat{f}_j = k\hat{f}_k - \omega\hat{f}_k \qquad \left(\hat{f}_k - \hat{f}_j\right)\omega = k\hat{f}_k - j\hat{f}_j \qquad \omega = \frac{k\hat{f}_k - j\hat{f}_j}{\hat{f}_k - \hat{f}_j}. \quad (201)$$

This allows us to measure peaks of fractional frequency in Fourier spectra. Fractional peaks are common, since the true signal is not synchronised to the window used for Fast Fourier Transform (FFT). When the peak amplitude is very close to the noise level, or is highly coincident with the FFT window, we necessarily do not detect more than one channel. In these cases, integer wavenumber is obtained, and a locking error may result. To avoid this, we will tests the signal with different signal windows for the FFT and pad the data with zeros as required.